SITUATION-AWARE SECURITY FOR WIRELESS AD HOC NETWORKS

BY

SEUNG YI

B.S., Seoul National University, 1995

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2005

Urbana, Illinois

To My Family

Abstract

Newly emerging ad hoc networking technology has enormous potential benefit for many mission-critical applications. However, providing adequate security required for the target applications presents many new challenges due to the unique nature of ad hoc environments. Complete decentralization and the absence of any support infrastructure in ad hoc environments effectively eliminates the possibility of the direct application of known security service designs from wired networks and infrastructure-based wireless networks. In this thesis, we investigate the environmental challenges posed by ad hoc environments and their effects on security service designs. Interactions between environmental factors, network properties and the quality of the security service are captured in a newly proposed concept called the *Situation*. Based on the understanding of situations, we provide design guidelines for *Situation-Aware Security* services where a security service is viewed as a best-effort service whose quality must be continuously monitored and quantified. The measurement of the varying quality of a security service is then conveyed to the end users to aid their decisions on appropriate usage of the provided security service. Based on this situation-aware security paradigm, we present two types of security services, security-aware routing for quantifiably secure route discovery and situation-aware key management that provides end users with the best-achievable authentication service with intuitive metrics to measure the provided quality of authentication.

Acknowledgments

There are numerous people that I should thank for helping me through the graduate program. First and foremost, I would like to thank my advisor Professor Robin Kravets for her continuing guidance and support. She was always available when I needed any kind of help and her constant endeavor for excellence has been a tremendous driving force for this work. I also thank the members of my committee, Professors Roy Campbell, Professor Klara Nahrstedt, and Professor Nitin Vaidya for their helpful comments and insights that made this thesis much deeper and broader. Professor Campbell deserves special thanks for his support and guidance during the first half of my graduate career. I'd also like to thank Professor Geneva Belford. She was my academic advisor when I first started in the graduate program and has been a constant support ever since. Anda Ohlsson was always there to help me navigate through the maze of departmental rules and regulations. Her help was invaluable.

Even though I often complained that I have not met enough good friends during my stay in Champaign, I now realize that it was simply not true. There are many people I am privileged to know. First, I would like thank my officemates in Mobius group, Albert Harris, Luiz Magalhaes, Cigdem Sengul, Yaling Yang, Robin Snader and Prashant Ratanchandani for countless interesting discussions and for attending countless practice talks and reading through numerous paper drafts without a complaint. I only wish that I was able to return the favor. I started my journey through the graduate program with three very good friends from System Software Research Group, Prasad Naldurg, Apu Kapadia, and Jalal Al-Muhtadi. We literally grew up together through many phases of the graduate school and I am glad that I had companions like you. I thank you all for being good friends. Outside of school, I owe a large part of my sanity to three years of Bushidokan training under my two wonderful senseis Albert and Erin Harris. I hope I could have a chance to repay the debt someday.

Last but not least, I'd like to thank my family; my parents for educating me with aspects from arts and sciences, for providing me the opportunity to pursue my dreams. My in-laws for unconditional support and encouragement to pursue my interests. Finally, I owe my deepest gratitude to my wife Aeri for for believing in me and for her everlasting love and support during this long journey.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CA Certificate Authority

CCV Chain Confidence Value

MoA Metrics of Authentication

PKI Public Key Infrastructure

QoA Quality of Authentication

Chapter 1

Introduction

Mobile ad hoc networking has emerged as an enabling technology to provide network support for challenging environments where providing network connectivity was previously infeasible or financially unjustified. Numerous situations, including battlefield communication support, emergency rescue operations, instant office meetings, school field trips, and city-wide wireless network connectivity via mesh networks, can readily and enormously benefit from ad hoc networking technology. Without relying on any stationary infrastructure, an ad hoc network is composed solely of mobile nodes communicating with one another using a wireless medium. An ad hoc network is a community-based and egalitarian way to create an instant network with two distinctive advantages in deployment over the traditional alternatives: *timeliness* and *low cost*. Since an ad hoc network does not require deployment of any heavy communication infrastructure, it can be deployed more quickly than wired or infrastructure-based wireless networks. Additionally, since an ad hoc network is formed among the mobile nodes at the scene instead of adding any additional communication infrastructure, deployment of an ad hoc network does not incur any extra hardware cost. However, the very characteristics that enable these advantages can act as a double-edged sword for providing the necessary security support, which is implicitly required by many applications served by ad hoc networks including battlefield communication and emergency rescue support. Security can easily be the single biggest obstacle to be solved before an ad hoc network can be deployed in reality.

Since its emergence in the early 90's, ad hoc networking has received a lot of attention from both the research community and commercial interests. Most of the early research was focused on solving the problem of routing packets using cooperative forwarding in this new environment [C. 99, J.]. Challenges in lower layers of the network protocol stack, management of data inside ad hoc networks

at the higher layers, and also cross-layer issues are some of the more recent interests of the research community. All of these efforts strive to handle the fundamental challenges of ad hoc environments in different layers with different approaches. Essentially, the goal is to provide the same quality and types of services as in traditional networks. By the same token, the goal of designing ad hoc security services is to provide the equivalent level of security support as in wired environments. In the context of providing security support, we recognize four environmental factors as the fundamental obstacles to providing strong security support in ad hoc environments: (1) vulnerable mobile nodes and heavy reliance on these insecure nodes for crucial network functions, (2) a public wireless medium prone to eavesdropping, (3) the absence of a reliable infrastructure, and (4) dynamic network state due to the mobility of nodes. Earlier approaches to supporting security in ad hoc networks typically solve some of these challenges but did not provide universal solutions. Recently, more unified approaches, including cross-layer solutions, have been proposed to solve all of these challenges at the same time. Unfortunately, most security research for ad hoc environments is still in its early stages where proposed solutions can only solve a part of the challenges under strong assumptions. The main reason for this phenomenon is the lack of a deeper understanding of the challenges. In other words, researchers have been trying to adopt known solutions from traditional network environments in radically different ad hoc environments without first fully understanding the vast differences. What is necessary is a significant change in view or a shift of paradigm that provides a new view of the security challenges as a whole.

Most previous efforts toward ad hoc network security attempting to adopt existing solutions from wired network environments capture only a subset of the differences while trying to provide an equivalent level of security, resulting in solutions that usually break under slightly changed or mildly stressful conditions. A more viable way to approach this problem is first to clearly identify the differences between ad hoc and traditional environments and then address these differences as an explicit part of the design process for ad hoc security services. In other words, the *hidden* and *implicit assumptions* that traditional security systems have relied on must be explicitly identified and exposed in the design of an ad hoc security services. Additionally, the impact of these assumptions on the quality of security services must be measured and these measurements must be used to decide the trade-off between the cost of security and the quality of security. In other words,

security must be understood as a best-effort service instead of a guaranteed service.

To this end, we propose a new approach to addressing these security challenges called *Situation-Aware Security*. First, we introduce the concept of *Situation*, which is the overall information and knowledge about a security service including all of its implicit assumptions. A *situation* for a security service consists of two components: the *design* of the specific security service and the end user's *knowledge* and *observations* about environmental impacts that can affect the vulnerability of the security service. Give this situational information, the quality of the available service can be measured and exposed to end users. This *Situation-Aware Security* paradigm is a more viable alternative to providing practical security support in ad hoc networks. The main challenges for providing situation-aware security include the following four points. First, an ad hoc security service must be designed to provide the *best-achievable* security service and to expose the relevant information about its design parameters at the same time. This information serves as the basis to form a *situation*. Second, the relevant set of *situational* information, including the design parameters and environmental effects, must be identified for each ad hoc security service. *(i.e., "What affects the quality of security").* Third, a concise and intuitive set of metrics must be designed to calculate the changing levels of the quality of security *(i.e., "How to measure the quality of security").* Lastly, all of this information must be presented to end users through an *intuitive interface* that can be easily understood. *(i.e., "How to present the measured quality of security").*

In Chapter 2, we present the concept of situation-aware security in detail. Then, we proceed with two types of security services designed based on the situation-aware security paradigm. In Chapter 3, we investigate the challenge stemming from the absence of a dedicated routing infrastructure or a transit network. Instead of communicating through a trusted transit network, mobile nodes in an ad hoc network communicate with one another relying on other ordinary mobile nodes. We identify newly presented challenges in this different environment and investigate subsequent vulnerabilities. We present *Security-Aware Routing (SAR)* to provide the concept of trustworthiness in ad hoc routing. In the following three chapters, we investigate the challenge of providing a reliable and available authentication services in ad hoc networks. Many ad hoc security services, including our own SAR, rely on other fundamental security services such as authentication. In Chapter 4, we present the goals and design principles of ad hoc key management frameworks that

provide authentication service and survey the existing approaches with our observations about why these approaches have failed to address the challenges adequately. As the solution, we propose *Situation-Aware Key Management*. As a part of Situation-Aware Key Management, we present an extensive trust model and concise metrics to measure the *Quality of Authentication (QoA)* using the wisdom from philosophy research called *Scientific Confirmation Theory*. In Chapter 5, we present the first incarnation of situation-aware key management in the form of a distributed public key infrastructure (PKI) design called MOCA. MOCA provides very high quality authentication service with support for measuring the changing quality of authentication depending on the network status. We further extend the design of MOCA in Chapter 6 to address the unique challenges from an ad hoc environment and present a novel way of seamlessly combining multiple authentication mechanisms called *Composite Key Management*. In Chapter 7, we apply the metrics for QoA to all existing key management framework designs and present the results from an extensive comparison study. The results from this study again demonstrate the strength and necessity of the situation-aware security paradigm. Finally, we conclude in Chapter 8 with several future research directions.

Under normal conditions the research scientist is not an innovator but a solver of puzzles, and the puzzles upon which he concentrates are just those which he believes can be both stated and solved within the existing scientific tradition.

Thomas Kuhn

Chapter 2

Situation-Aware Security

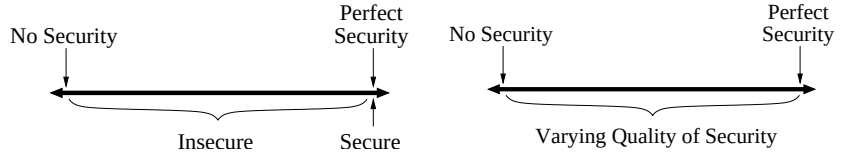
As a technology matures, many assumptions and requirements that are initially investigated become widely accepted across the research community and eventually they become invisible and hidden, and eventually forgotten. This is the natural evolution of a research area and enables researchers to focus their attention on more relevant issues and deepen their understanding based on such abstractions. However, every so often there comes a time when all such forgotten assumptions need to be reexamined due to radical changes in environments or significant new findings that alter the accepted view. We have observed a good example of such a situation with the advent of wireless networking. During the last three decades, the main thrust of networking research has been the Internet. After two decades of research, networking technology for wired networks was mature when wireless technology started showing up. However, much of the wisdom learned through Internet research efforts could not be directly applied to the new wireless environments because this new environment violated some of the key assumptions that were made and subsequently forgotten for the wired network. Perhaps the best example is the use of TCP (Transmission Control Protocol) in wireless networks [BSAK95, HV02, LK00, MCG$^+$01]. TCP was designed upon a very strong assumption that a packet loss is only caused by congestion inside the network [Pos81]. Therefore, a packet loss causes the TCP sender to back off and reduce its load on the network, alleviating congestion. While this assumption is indeed strong, it holds true in wired networks such as the Internet and served well in the refinement of TCP. Even though TCP has been studied and enhanced extensively over the years, it did not work as expected when first adopted to wireless environments. The culprit was the fact that the wireless environments introduced another cause for the packet loss: wireless channel errors. This seemingly unimportant difference in the new

environment practically broke the well-refined TCP in wireless networks, resulting in very poor performance results. Once this problem was recognized, many variations of TCP were proposed, most of which try to distinguish between congestion losses and channel losses. In other words, TCP had a *hidden assumption* that did not hold anymore in the new environment and exposing the hidden assumption was necessary for TCP to work in a wireless setting.

Similar observations can be made about providing security support for ad hoc environments. Security research in wired networks and infrastructure-based wireless networks such as cellular networks have been well investigated over the years. However, newly emerging ad hoc environments present challenges that require us to re-examine many of the forgotten assumptions that were irrelevant in previous settings. For example, wired/cellular network environments enjoy a set of luxuries that do not exist in ad hoc networks, such as reliable connectivity, stable network routes, and availability of secure and reliable servers. Naturally, most security systems designed for wired networks are designed to maximize the bounty of these facilities. As a result, many existing security service designs rely heavily on the availability of a secure and fault-tolerant trusted third party and stable connectivity to it. As ad hoc networks became more popular, many researchers attempted to adopt solutions from the Internet into ad hoc environments without examining the potential *hidden* assumptions. Essentially, they tried to address the challenges in ad hoc environments without first clearly identifying them, resulting in incomplete or ineffective designs. As a result, these systems often break under conditions that are slightly different or more stressful than what they were designed for. Furthermore, since these assumptions were deeply integrated into the design of the solutions, it is unclear when they will break and why they do. A better approach would be first to identify and recognize the impacts of ad hoc environments on security services and then to design security services with clearly stated assumptions or dependencies on the environment. In other words, the hidden assumptions of existing security service designs must be *exposed*. Furthermore, simply recognizing the difference is not enough because of the unstable and volatile nature of dynamic ad hoc environments. We argue that the effects of these hidden assumptions must be quantified because the effects can be significant and change rapidly. Therefore, the effects of environmental factors on the hidden assumptions must be constantly monitored and conveyed to end users.

2.1 Security as a Spectrum

Traditionally, we are accustomed to thinking of security as an absolute concept or a guaranteed service. In other words, the main design goal of security services has been to provide a perfect level of security. For example, a firewall is designed to block all unauthorized accesses and not some portion of them. Likewise, an encryption algorithm is expected to generate truly random-looking ciphertext and not somewhat random output. However, realizing this seemingly natural goal is not a trivial matter. Moreover, in ad hoc environments where network conditions can change very rapidly, it is almost impossible to design a security service that is immune to all environmental effects of ad hoc networks.



(a) Traditional View on Security (b) Situation-Aware Security

Figure 2.1: Two Views on Understanding Security

If we visualize the quality of a security service as a continuous spectrum with no security at one end and perfect security at the other end, the traditional view is to consider the single extreme point of perfect security, or a very narrow margin around that single point, as acceptable (Figure 2.1 (a)). Designing a security service to achieve this strict goal in any complex system may be possible in theory, but almost impossible in reality due to bugs, excessive costs, environmental difficulties, or the difficulty in managing the sheer complexity of the system. There have recently been attempts to address these problems by designing systems to expect and tolerate minor malfunctions or intrusions [CLR+02, PBB+02]. We believe this is the right direction to approaching security challenges in any complex system.

However, in ad hoc environments, it is almost impossible to maintain a security service to provide even almost perfect security due to the challenging environmental conditions. Therefore, we propose to extend the previous view of tolerating minor imperfections even further to the point where a security service is considered as a best-effort service with changing quality over time (Figure 2.1 (b)). Instead of treating faults and compromises as rare exceptions to handle,

they must be viewed as a part of the normal operating conditions that affect the provided level of service. Instead of trying to design a perfect security service in ad hoc networks, which is likely to be infeasible or too expensive to maintain, a security service can be designed to provide the best-achievable service under varying conditions. The consequences of this change in view forces end users of such services to change their perception of security from an absolute binary concept with only two states {*secure, insecure*} to a service with continuous levels. Therefore, end users of such security services must be equipped with the facility to *measure* the quality of the provided service. Based on such measurements, end users can make well-informed decisions whether to utilize the security service at its current level or to wait until a higher level of service becomes available, or even employ additional means to improve upon currently available security services.

Environmental factors in ad hoc networks that affect the quality of available security can fluctuate rapidly and to a significant extent. Since the level of an available security service can also change rapidly as a result, the measurement of quality of security must also be performed frequently. In the remainder of this chapter, we first discuss the unique environmental challenges in ad hoc networks that impact security services in the following section. Then, we propose the concept of *Situation* that captures all relevant environmental factors for a given security service. We conclude this chapter with a survey of research efforts related to the situation-aware security paradigm.

2.2 Environmental Factors in Ad Hoc Networks

The unique infrastructure-free nature of ad hoc networks results in several distinct characteristics that are not found in traditional networks and we identify four such factors that can affect security services.

First, all nodes in an ad hoc network are assumed to be mobile end users. Therefore, the physical security of each node is seriously degraded since it is much easier to compromise a mobile node that is prone to capture. Moreover, due to the absence of any support infrastructure, all critical network functions must be performed collectively within the network. The most obvious example of this is found in the design of ad hoc routing protocols [J. , R. 97, Y. 98, V. 97, C. 99, E. 99, C-K97] . Unlike wired networks where a small number of service providers "own" the transit network, an ad hoc network is collectively owned by the mobile nodes that form the network. Therefore, the

network infrastructure can no longer be implicitly trusted as it is in the Internet.

Second, all communication in an ad hoc network uses a wireless medium, which is inherently shared, and controlling access to the wireless medium is not simple. The best showcase of this is the jamming [XTZW05] attack. Jamming is an attack at the physical layer where a malicious node can emit powerful signals that overshadow all meaningful and legitimate communication in an area. Unless access to the physical area can be controlled, jamming cannot be defeated. Therefore, any service for ad hoc environments must be designed to handle such network connectivity disruptions. A less severe case of an attack against the wireless medium is eavesdropping where any node with a wireless receiver can capture the communication traffic. Therefore, an additional level of protection is required for sensitive data traffic required for an ad hoc security service.

Third, by definition, an ad hoc network does not rely on a special-purpose communication infrastructure, such as routers in the Internet or base stations in cellular networks. This causes problems in attempts to reuse existing approaches for wired network security in ad hoc networks [SMGLA96, SNS88]. Most security solutions designed for wired networks assume that it is possible to maintain secure and reliable servers against malicious attacks and random faults altogether. For example, PKI [KP] for wired networks is designed to rely on secure hosts that serve as the certificate authority. However, the same design cannot be applied to ad hoc environments due to the inherent physical vulnerability of mobile nodes. Therefore, the design of an ad hoc security service must be approached in a different way by relying on the collective provision of security services.

Finally, network connectivity can change rapidly in ad hoc networks due to the mobility of nodes. Existing security solutions for wired networks typically assume that the network is almost always fully connected [SNS88]. That is, any node in a network can expect to communicate with any other node without disruption. However, network connectivity in ad hoc networks is a function of mobility and cannot be guaranteed unless the mobility of nodes is controlled. For example, a popular way to authenticate a mobile user in a wireless LAN environment is "Wi-Fi Protected Access (WPA)", which relies on a back-end authentication server [All]. In WPA, a mobile node is presented with a challenge and its response is verified by the authentication server located inside the network. The connectivity to this authentication server must be guaranteed. Otherwise,

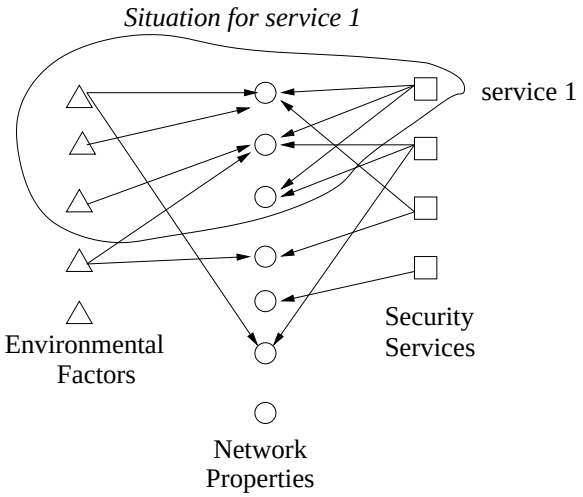


Figure 2.2: Dependency Relationships

mobile users can be randomly denied network access. This approach cannot be adopted in ad hoc environments since it is not possible to guarantee connectivity to any certain node at any given time. In summary, an ad hoc security service must be designed without any strong assumptions about stable network connectivity.

These four unique challenges in ad hoc environments affect the performance of security services, typically degrading the quality of service provided by them. To capture these service-specific effects of environmental factors, we next define a new concept of *Situation*.

2.3 Capturing Environmental Impacts with the *Situation*

The effects of environmental factors on security services are specific to the design of each service and the assumptions it makes. We capture these service-specific environmental effects in the concept of *Situation*. A *situation* is a set of hidden assumptions that a specific security service relies on and environmental factors that can affect such hidden assumptions. The relationships between the hidden assumptions and environmental factors can be captured by mapping the security services, network properties and environmental factors using a graph structure. In Figure 2.2, the square objects in the right column represent different *security services*. These services each rely on certain *network properties* to operate. The network properties are represented as the circular objects in the middle column. Edges from a service object to a network property indicate that the service relies on

the specific network property. In the left column, triangles represent the *environmental factors* that exist in ad hoc networks, including the ones discussed in Section 2.2. Each of these environmental factors affect a set of network properties, typically resulting in the fluctuation of the properties, as denoted by the edges from the environmental factors to the network properties. Given a service, *service 1*, at the top of the right column, we can easily identify the network properties this service relies on and, in turn, the environmental effects that can degrade these network properties. We define the concept that includes all of these players as the *situation*, which is essentially the set of environmental factors that can degrade the network properties required by a specific service.

In traditional wired network environments, the effects of environmental factors are negligible. Therefore, there was no need to expose the interactions between the environmental factors and the relevant network properties for a wired security service. In comparison, in ad hoc environments, the impacts of environmental factors in the left column can be significant and also frequently changing, resulting in rapid fluctuation of the quality of security provided by security services. Hence, we argue for two points. First, all hidden and forgotten assumptions need to be made explicit and the degradation of each assumption must be clearly understood. Second, since the environmental factors in ad hoc environments can change frequently, the effects of the changing network properties are also frequent and significant, resulting in potentially severe fluctuations in the achievable level of security by the services. Therefore, monitoring must be constantly provided and measurements must be continuously provided to end users.

However, it is not wise to provide end users with the *raw* information about environmental effects on a specific security service. The information must be processed and tailored so that an end user can intuitively understand and act upon the information. In situation-aware security, the situational information is translated into the form of the *Quality of Security* before being conveyed to the end users. End users do not need to know the detailed situational interactions of a security service but only the changing quality of security resulting from such interactions. When an ad hoc security service is immune to any environmental effects, end users will get a perception of perfect quality of security. If environmental factors affect the security service, such a situation is translated into degraded quality of security. The specific meaning of quality of security is dependent on each security service and we will address two specific definitions for routing and for key management. In

the rest of this chapter, we examine related research efforts that share certain characteristics with the situation-aware security paradigm.

2.4 Related Research to Situation-Aware Security

We can categorize the related research for situation-aware security into two classes. First, there are various types of security services proposed for ad hoc environments. Most of these approaches *implicitly* include some notion of the *situation* concept in their design [CBH03, K. 02]. Most security services are designed based on a set of assumptions about relevant environmental factors and their impacts. However, none of the current approaches provide the monitoring functionality for the quality of the provided security service that is necessary to provide complete situation-awareness. Instead, they are designed to provide the intended high quality of security within a certain range of assumptions and fail when these assumptions are violated, resulting in silent failures. Since the network dynamics of an ad hoc environments can fluctuate significantly, designing an ad hoc security service that does not fail is either technically infeasible or prohibitively expensive, limiting the utility of these approaches. The second category of related research exists in various efforts to quantify or measure the quality of security. However, most previous research focuses on the trade-off between the cost and the quality of security in the interest of reducing costs when it is possible to provide near-perfect quality of security [Chu02, ONY03]. Although it may be possible to provide very good quality of security, some applications may not require such high levels of security. In such cases, it is desirable to reduce the quality of security to save the related costs while maintaining the required level of security. In other words, the quality of security is used to pinpoint the best trade-off point, not to provide changing information to end users in these research efforts. In comparison, we deal with ad hoc environments where providing a high level of security itself is challenging and the goal of situation-aware security is to accurately measure the best-achievable quality of security available from the security service under challenging network conditions. We discuss both categories of research efforts in detail in the remainder of this chapter.

2.4.1 Security for Ad Hoc Environments

As ad hoc networking technology receives more attention, security support for ad hoc environments has become a critical challenge. Numerous approaches have been proposed for various types of security services. However, none of the existing efforts for ad hoc network security have been designed following the situation-aware security paradigm. Each approach is designed based on a set of assumptions. As long as such assumptions are satisfied, the approach can satisfy the intended level of security service. However, once any of the assumptions is violated, these approaches fail silently without providing any useful information. Considering the fact that it is extremely difficult to design an ad hoc security service that does not fail under any condition, an ad hoc security service failing without any warning or useful information is not desirable. Situation-aware security addresses this point by not classifying the performance of an ad hoc security service as success or failure, but to treat it as continuous levels of service.

Secure routing and related solutions were the biggest focus of the community in the beginning. Hu et al._propose several variations of secure routing protocols including a proactive approach [HJP02] and a reactive approach [HPJ02], each with a certain set of assumptions, such as support of public key cryptography and no denial of service attacks. However, since these approaches do not tolerate violations of the assumptions, when these assumptions fail, both of these approaches fail. In other words, these approaches may not work within the current situation of the network. As a first step in the direction of the situation-aware security Dahill et al._suggest classifying ad hoc networks into four categories using two criteria: open vs._managed and friendly vs._hostile [K. 02]. However, they do not provide any measurement capability in their secure routing protocol called ARAN. Therefore, ARAN also silently fails when the assumptions of the design are violated. Capkun et al._propose another version of secure routing that does not rely on any strong key management support [CH03] based on a different set of assumptions without relying on public key cryptography. Some novel attacks on ad hoc routing protocols are also discovered and discussed. Hu et al. first propose a new type of attack called "rushing attacks" on ad hoc routing protocols and propose the Packet Leash protocol to prevent such attacks [HPJ03a] under the assumption that the verifying node either has geographical information or clocks on mobile nodes are loosely synchronized. While all of these secure routing protocols can ensure the correct

behavior of the routing protocols under the respective assumptions, none of them is equipped with the capability to monitor the behavior of the protocols or to detect the violation of the correct behavior. Essentially, they can satisfy the goals as long as their assumptions hold but do not provide any guarantees once the assumptions are violated. Furthermore, if the assumptions fail, there is no means to measure what kind of impact such failure causes.

Related to secure routing research is the incentive system for mobile nodes. Packets in ad hoc networks are forwarded by intermediate mobile nodes whose interests do not necessarily lie in helping each other. Several approaches have been proposed to give proper incentives to "selfish" mobile nodes so that enough participation can be "incentivized". Marti et al._present the first attempt where each mobile node is equipped with the ability to detect their neighboring nodes' behavior and penalize selfish nodes [S. 00] assuming that mobile nodes are capable of promiscuous eavesdropping. Buchegger et al._propose the CONFIDANT protocol that enforces fair cooperation among mobile nodes in an ad hoc network [BB02]. Buttyan et al._propose a virtual currency based scheme for packet forwarding where forwarding nodes get paid for their labor based on the assumption that mobile nodes are equipped with tamper-proof hardware [BH03]. Again, all of these approaches will function correctly as long as their respective assumptions are satisfied. However, it is unclear how their utility will degrade once the assumptions are violated.

Cryptographic key management also has received a fair amount of attention from the research community [CBH03, CHB03, HBC01, KZL$^+$01, YK03, ZH99]. However, previous research shows similar traits to the other areas. They are all designed to function effectively under a certain set of *reasonable* assumptions. However, it is unclear what constitutes a *reasonable assumption* in ad hoc environments and none of the proposed designs deal with situations when their assumptions are violated. We will discuss these issues for key management in more detail in Chapters 5 and 6 where we present our situation-aware key management frameworks.

2.4.2 Measuring the Quality of Security

The main purpose of current efforts to define and measure the quality of security services under various conditions. The main purpose of these efforts is to find the right trade-off point where good-enough security can be provided in a cost-effective way. In other words, the quality of security is

treated as a tunable parameter selected by either end users or network operators. Tunable security is only meaningful when it is possible to have a perfect security even at a high price. Then, the goal of *tuning* the security level is to reduce the cost while maintaining the necessary level of security. While the situation-aware security paradigm also measures the quality of a security service, the situation-aware paradigm differs in one critical aspect. The primary reason for monitoring and measuring the quality of security service in situation-aware security is not to tune the quality of the service but to accurately measure rapidly the changing quality of the best-achievable security under the challenging network conditions and convey the measurements to end users for their benefit. This difference is based on the fact that end users have little or no control over an ad hoc network. Therefore, end users cannot tune many parameters but instead must adapt to the currently available levels of service.

Liu et al._proposed a metric called "Quality of Protection" that quantifies the level of protection for critical operating system services in active network environments [CLM+99]. Ong et al. also used the term "Quality of Protection" to quantify the level of protection for multimedia data depending on the types of devices, users and requirements for data traffic [ONY03]. Chu performed an interesting study in his master's thesis where he designed a framework to turn off some of the security services in trusted operating systems for improved performance [Chu02]. In his thesis, Chu suggested that not all applications require the strong security support available from the underlying trusted operating system and that some applications can benefit from turning off unnecessary security support to achieve better performance. Chu observed that there is a trade-off between the level of security and the performance of applications. Ong and Chu's schemes share the notion that the quality for security and the required cost should be a tunable parameter that end users or applications can change. A similar trade-off observation is made in Lindskog's work where he treated the level of encryption as the tunable parameter where the grade of encryption can be decreased to achieve better performance in resource-constrained devices [LSHJ04]. Lindskog expanded his research to general security and the results presented in his PhD thesis [Lin05] are perhaps the closest result to this thesis in a sense that he also views the security as a continuous spectrum and not a binary state and tries to measure the amount of security achievable. While Lindskog's approach focuses on the effect of resource-constrained devices, situation-aware security

can address all possible network conditions including the effects of resource-constrained devices.

In the next chapter, we introduce the first application of the situation-aware security paradigm to ad hoc routing, the *Security-Aware Routing (SAR)*. Instead of blindly treating the intermediate nodes as a trustworthy transit network, SAR quantifies the amount of trust provided by a network route based on the trustworthiness of participating nodes and their capability to handle sensitive security operations. SAR enables end users to discover trustworthy routes that can be used to transmit sensitive traffic. Many ad hoc security services such as SAR require a reliable and secure authentication service to function. Next, we apply the situation-aware security paradigm to key management services. A well-established way to provide authentication service in dynamic environments is to deploy a cryptographic key management framework. In Chapter 4, we lay out the groundwork for our approach to situation-aware key management frameworks, followed by the MOCA distributed PKI framework in Chapter 5 and Composite Key Management in Chapter 6. In Chapter 7, we present an extensive comparison study of existing ad hoc key management frameworks and expose newly discovered design flaws stemming from the lack of situation-awareness in their designs.

Chapter 3

Security-Aware Routing

Most ad hoc routing protocols are cooperative by nature, depending on neighboring nodes for packet forwarding [E. 99]. While such cooperative packet forwarding appears to be a natural solution in ad hoc environments, it relies on one very critical hidden assumption: *trust in the transit network*. In a wired network like the Internet or an infrastructure-based wireless network like cellular networks or wireless LANs, there is implicit trust put upon the transit network between two communicating parties since there is a clear distinction between *clients* and *providers* of the network support. For example, Internet service providers provide the transit network for Internet communication. Similarly, cellular carriers and hosting organizations provide the transit network for cellular communication and wireless LAN communication. Due to this clear distinction between the users of the network and the providers of the network, it is in the providers' best interest to provide a reliable service and clients can have reasonable expectation and trust in the behavior of the transit networks. However, such assumptions do not hold anymore in ad hoc networks since there is no separate transit network. Packets are forwarded by other mobile nodes whose main interest is not in providing service to others. As a result, all ad hoc routing protocols become vulnerable to malicious intermediate nodes. This naïve trust model allows malicious nodes to paralyze an ad hoc network by inserting erroneous routing updates, replaying old routing information, changing routing updates, advertising incorrect routing information, dropping packets, and also looking inside the packets.

While these attacks are possible in fixed networks as well, the nature of the ad hoc environment magnifies their effects, and makes their detection difficult [Y. 00b]. Secure routing solves some of these challenges by enforcing that mobile nodes follow the protocol specification. However, a

secure routing protocol cannot prevent any attacks on data traffic, such as dropping packets and eavesdropping, as long as the malicious nodes adhere to the behaviors specified in the protocol. Many approaches [BH03, BB02, S. 00] have been proposed to provide incentives to forwarding nodes so that they do not drop packets but none has been shown to provide a universal solution. Not much attention has been paid to eavesdropping or peeking into the packets being forwarded by the forwarding nodes since most researchers operate on the wrong assumption of a trustworthy transit network.

This distinct absence of a trustworthy transit network in ad hoc environments demands a new routing metric that can adequately address this new threat. Traditionally, distance (measured in hops) is used as the metric in most ad hoc route-discovery algorithms [J. , V. 97, C. 99, Z. 97]. The use of other metrics (e.g., geographic location, signal stability, power, load on nodes etc. [R. 97, Y. 98, S. 98]) can improve the quality and the relevance of the routes discovered for particular applications and configurations. Along these lines, we explore the use of different security attributes to improve the quality of the security of an ad hoc route. In this chapter, we present *Security-Aware ad-hoc Routing (SAR)*, an approach to ad hoc routing that incorporates the security levels of nodes into traditional routing metrics. Our goal is to characterize and explicitly represent the trust values and trust relationships associated with ad hoc nodes and use these values to make routing decisions. SAR essentially creates virtual overlays for routing, where the characteristics of the overlays are defined by the security requirements of the application and the topology of the overlay is defined by the capabilities of the mobile nodes. We quantify the notion of trust and represent the trust relationships explicitly by defining a suitable hierarchy of trust values. Furthermore, this trust measurements are fed back to end users so that they can have a clear idea about the quality of routes discovered by SAR, providing *situation-awareness* to end users. In effect, SAR enables end users to discover secure overlays that exist in an ad hoc network.

Ensuring that data is routed through a trustworthy route composed of trusted nodes must be accompanied with support of a secure routing protocol that can enforce the correct behavior of SAR. SAR is an approach that provides secure and trustworthy routes for *data packets* and still requires secure routing support for its *control packets*, which can be integrated to a SAR protocol or based on any external secure routing approaches.

The trust value of a node and the security attributes of a route are intimately connected in our framework to provide a unified view of security. We introduce the notion of an integrated security metric that is a combination of security attributes and trust levels. We augment existing ad hoc routing algorithms with this integrated metric to steer route discovery and route maintenance behavior. Our route discovery mechanism finds nodes that match particular security attributes and trust levels. Only nodes that provide the required level of security can generate or propagate route requests, updates, or replies, defining the virtual overlay for secure communication. By restricting communication to such capable nodes, SAR generates fewer routing protocol messages. Furthermore, security policies can be encoded into the attributes to enable policy-based secure routing.

The rest of the chapter is organized as follows: In Section 3.1, we present our motivation with a representative example, expand on the characteristics of an ad hoc network that make it vulnerable to routing attacks and briefly describe a threat model. In Section 3.2, we present our generalized SAR protocol for quantifiable secure route discovery, update, and propagation with trust levels and security attributes as metrics. This section includes related research, a description of the traditional definitions and metrics of routing protocol security, and outlines a mechanism to quantify and measure the protection associated with particular routing protocol incarnations. In Section 3.3, we revisit our threat model, develop an attack classification and validate our protocol against this model. Section 3.4 describes our experimental test bed and our modifications to AODV to enable security-aware routing. Section 3.5 presents the performance evaluation of our prototype in detail. Finally, Section 3.6 presents our conclusions.

3.1 Motivation

In this section, we motivate the need for security awareness in ad hoc networks at the routing level with a battlefield communication scenario along with the application of situation-awareness to the problem.

3.1.1 Example Scenario

To illustrate the need for network level security awareness, we present an example scenario where finding a route with specific security attributes or trust levels is more relevant than finding the shortest route (or any route) between two nodes. We focus on a high-risk ad hoc network, wireless communication devices in a battlefield, where malicious adversaries can intercept and alter mission critical information.

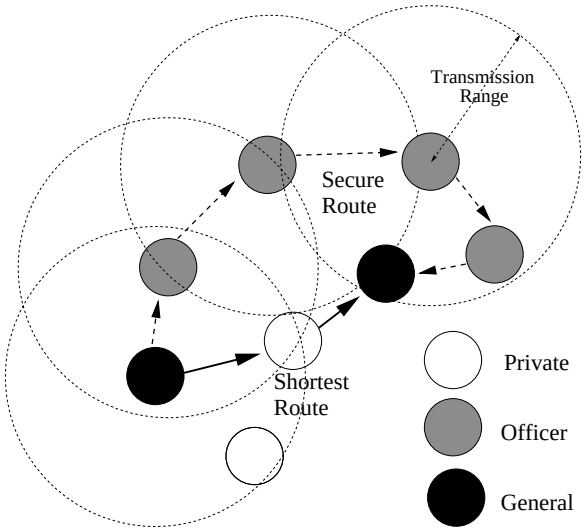


Figure 3.1: Security-aware Routing - Motivation

Consider the scenario where two generals establish a route to communicate among themselves, using a generic on-demand ad hoc routing protocol(see Figure 3.1). During the mission, the generals detect that some of the privates have defected. Relaying these messages using potentially compromised nodes can leak information to untrusted entities and jeopardize the mission. Even if the generals encrypt the information flowing between them, the fact that they are communicating may disclose that a strike is imminent. Another threat could be that traitors may be able to store the messages or send them to enemy nodes for cryptanalysis. Therefore, the generals decide that they can only trust nodes owned by officers to route their packets. Security-aware routing enables the generals to route around the problem nodes and establish an alternate route with greater security guarantees. The sending general's route discovery protocol embeds the rank of the node as a metric in its negotiation and tries to establish a route that avoids all privates. If such a route exists, as shown in the figure, a route passing through only the officers can be set up. If no route with the

required security attributes or "quality of protection" exists, a notification to the sender can allow re-negotiation. Based on such feedback, for example, the generals may decide to set up a route that can support 128-bit encryption, knowing that the privates cannot cryptanalyze or transmit the messages very far with their inferior devices.

From this example, we observe that senders can make informed decisions about the amount of trust that can be put upon a network route available to their data packets throughout the route by embedding security attributes into the route discovery protocol itself. Furthermore, this quality of protection offered by the route directly affects the security of the data packets exchanged between the nodes on a particular route. Route updates and route propagation messages are also protected by this technique. In the next subsection, we enumerate the characteristics of the ad hoc network environment that make them vulnerable to misbehaving forwarders. To strengthen our motivation to include security as a fundamental attribute or metric in ad hoc routing protocols, we also present a threat model. This model enumerates the vulnerabilities and threats that expose the communication of routing protocol packets among nodes in an ad hoc network to malicious attackers.

3.1.2 Untrusted Transit Network: The Hidden Assumption in Ad Hoc Routing

Ad hoc wireless routing protocols assume that the mobile nodes are cooperating with each other to route packets from their source to their destination. Routing protocol packets carry important control information that governs the behavior of data transmission in the ad hoc network. Without adequate protection, these packets can be easily subverted or modified. A secure routing protocol is required to protect control packets and an extensive amount of research has been presented on the topic [CH03, HJP02, HPJ02, K. 02, S. , SMGLA96, ZH99]. However, a secure routing protocol can only guarantee the correct behavior of mobile nodes to discover, repair, and salvage routes but cannot protect the following data traffic that goes through the discovered routes. In other words, a secure routing protocol ensures that all nodes follow the protocol specifications but is oblivious to what these specifications should contain. Since the level of trust in a traditional ad hoc network cannot be measured or enforced, enemy nodes or compromised nodes may participate

correctly in route discovery and then intercept and filter the following data packets to disrupt communication. Compromised users may use the information gleaned from transit packets to mount an attack or anticipate combat moves to their advantage. Since there is no penalty or punishment for such misbehavior, in general, nodes have no incentive to behave well, as long as they follow the specifications of the routing protocol.

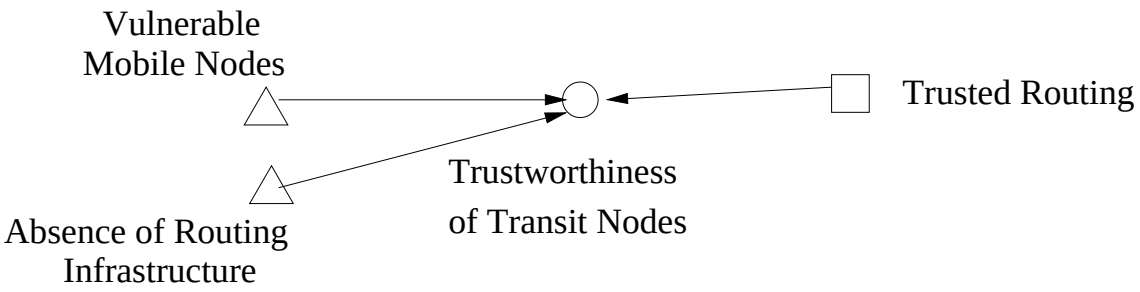


Figure 3.2: Situation for Security-aware Routing

A trustworthy routing service relies on the network property of the trustworthiness of transit nodes. This network property is affected by two changing environmental factors: physical vulnerability of the transit nodes and the absence of specialized routing infrastructure. By quantifying the trustworthiness of participating nodes, the trustworthiness of a network route can be effectively captured. Based on these observations, we define the *situation* for trustworthy ad hoc routing protocols as in Figure 3.2.

3.2 Security-Aware ad hoc Routing (SAR)

Most ad hoc routing protocols were designed as modifications or augmentations to traditional routing protocols for wired networks [C. 94]. These protocols send updates and react to topology changes, using monitoring and other infrastructure support to maintain routing tables. Current research focuses on pure on-demand [J. , C. 99] routing protocols, and more recently, on augmentations that exploit additional information available on the ad hoc nodes [Y. 98, R. 97, S. 98] to improve the quality of routes and reduce performance overheads.

Most of the protocols that have been proposed so far focus on discovering the shortest path between two nodes as fast as possible. In other words, the length of the routes is the only metric used in these protocols. Some protocols trade performance and simplified management to obtain

bounded sub-optimal paths to speed up the route discovery process [Z. 97, P. 99]. However, the protocol metric is still the length of the routes, measured typically as hop-count. In this chapter, we contend that that there are applications that require more than just the assurance that their route has the shortest length. We argue that end users must be able to specify the required level of trustworthiness for their routes with respect to metrics that are relevant to them. Routes that satisfy these requirements may or may not exist depending on the changing network conditions and such changing *situation* must be conveyed to the end users so that they can adequately adapt their behavior. Our approach shares certain similarity with the policy based routing protocols for QoS [E.].

3.2.1 Protocol

For simplicity, we assume that the base protocol is an on-demand protocol similar to AODV [C. 99] or DSR [J.]. In the original protocol, when a node wants to communicate with another node, it broadcasts a Route Request or RREQ packet to its neighbors. The RREQ is propagated to neighbors of neighbors and so on, using controlled flooding. The RREQ packets set up a reverse path to the source of the RREQ on intermediate routers that forward this packet. If any intermediate node already has a path to the RREQ destination, this intermediate node replies with a Route Reply or RREP packet, using the reverse path to the source. Otherwise, if there exists a route (or connectivity) in the ad hoc network, the RREQ packet will eventually reach the intended destination. The destination node generates a RREP packet, and the reverse path is used to set up a route in the forward direction (RPF or Reverse Path Forwarding).

In SAR, we embed our security metric into the RREQ packet itself, and change the forwarding behavior of the protocol with respect to RREQs. Intermediate nodes receive an RREQ packet with a particular security metric or trust level. SAR ensures that this node can only process the packet or forward it if the node itself can provide the required security or has the required authorization or trust level. If the node cannot provide the required security, the RREQ is dropped. If an end-to-end path with the required security attributes can be found, a suitably modified RREP is sent from an intermediate node or the eventual destination. SAR can be implemented based on any on-demand ad hoc routing protocol with suitable modifications. In this chapter, we use AODV [C. 99] as our

platform to implement SAR.

3.2.2 Behavior

Our modification to the traditional ad hoc routing protocol changes the nature of the routes discovered in an ad hoc network. The route discovered by SAR between two communicating entities may not be the shortest route in terms of hop-count. However, SAR is able to find a route with a quantifiable guarantee of security. If one or more routes that satisfy the required security attributes exist, SAR will find the shortest such route. If all of the nodes on the shortest path (in terms of hop count) between two nodes can satisfy the security requirements, SAR will find routes that are optimal. However, if the ad hoc network does not have a path with nodes that meet the RREQ's security requirements, SAR may fail to find a route even if the network is connected.

3.2.3 Protocol Metrics

In this subsection, we enumerate different techniques to measure or specify the quality of security of a route discovered by our generalized SAR protocol. The first technique is the explicit representation of trust levels using a simple hierarchy that reflects organizational privileges. The second is the use of security capability of participating nodes. The next subsection enumerates the different techniques used to protect the integrity of routing messages.

Trust Hierarchy

SAR provides applications the ability to incorporate explicit trust levels into the route discovery process. Most organizations have an internal hierarchy of privileges. For example, in our battlefield scenario, the military ranks of the users of the ad hoc nodes form an explicit partial-ordering of privilege levels. A simple way of incorporating trust levels into ad hoc networks is to mirror the organizational hierarchy, and associate a number with each privilege level. These numbers represent the security/importance/capability of the mobile nodes and also of the routes. Simple comparison operators can sort these levels to reflect their position in the actual hierarchy. This closely follows research in information flow theory [SV98] and mandatory access control (MAC) [BP75]. In information flow theory, a piece of information can only flow into a variable of equal or higher security

level. Similarly, in mandatory access control schemes, an object can be only accessed by subjects of equal or higher security clearance. SAR mirrors such structure by viewing data packets as the objects to be protected and intermediate nodes as the subjects that try to access the objects.

Security Capabilities

In addition to the ranks of the mobile nodes, SAR incorporates the nodes' capability to perform necessary security operations. The list of available operations will vary depending on the deployment but typically include capability to create and verify digital signatures, encrypt and decrypt data packets and other relevant security operations. By explicitly including the security capabilities as a routing criteria, a source node can choose routes composed with mobile nodes that are adequately equipped for the necessary security support. The required set of security operations is set in a *security attribute bit vector* inside the route request message.

Protecting the Protocol Metric Requirement

It is important to note that that both trust level and security capability metrics must be immutable. A node with a lower trust level must not be able to arbitrarily change its trust level or change the trust level of the RREQ request it forwards. To provide this guarantee, many techniques can be employed. If keys can be distributed *a priori* or a key agreement can be reached by some form of authentication, the simplest technique is to encrypt the portion of the RREQ and RREP headers that contain the trust level. If all of the nodes in a trust level share a key, any node that does not belong to this level cannot decrypt or process the packet and is forced to drop it. If a node is compromised, tamper-proofing can prevent attackers from learning the values of the keys. In the design of SAR, we leverage related research in key management for ad hoc networks and assume that some mechanism to distribute keys and shared secrets, such as our own situation-aware key management are already in place.

3.3 Protection

In this section, we develop an attack classification and itemize the protection offered by SAR against attacks on the trust hierarchy and the information in transit in the routing protocol messages. Other

attacks on ad hoc networks and related solutions are also briefly discussed.

3.3.1 Trust levels

Attacks on the trust hierarchy can be broadly classified as *Outsider Attacks* and *Insider Attacks*, based on the trust value associated with the *identity* or the source of the attack. SAR modifies the behavior of route discovery, tying in protocol behavior with the trust level of a user. What is also needed is a binding between the identity of the user with the associated trust level. Without this binding, any user can impersonate anybody else and obtain the privileges associated with higher trust levels. To prevent this, stronger access control mechanisms are required (AAA or Authentication, Authorization and Accounting). To force the nodes and users to respect the trust hierarchy, cryptographic techniques such as encryption, public key certificates, shared secrets etc., can be employed. For example, all authenticated users belonging to a trust level can share a secret key.

Traditionally strong authentication schemes have been used to combat outsider attacks. The identity of a user is certified by a centralized authority and can be verified using a simple challenge-response protocol. Various schemes including the application of threshold cryptography, techniques for key sharing [AG00, BSSW02, CHB03, HBC01, KZL+01, ZH99], and techniques for key agreement between multiple cooperating entities in dynamic collaborative groups [Y. 00a] have been proposed to tackle the lack of a centralized authority in an ad hoc network. Our open design allows us to incorporate any of these mechanisms. For example, if one key is used per level, the trust levels are immutable and the trust hierarchy can be enforced. In our implementation, for simplicity, we use a simple shared secret to generate a symmetric encryption/decryption key per trust level. Packets are encrypted using this key and nodes and users belonging to different levels cannot even read the RREQ or RREP packets. Any user or node that is an outsider cannot obtain this key.

Insider attacks are launched by compromised users within a protection domain or trust level. The users may be behaving maliciously, or their identity may be compromised (key is broken etc.). However, insider attacks are generally contained within the compromised node's trust level. Routing protocol packets in existing ad hoc algorithms do not carry authenticated identities or authorization credentials and compromised nodes can potentially cause a lot of damage. Insider

attacks are hard to prevent in general at the protocol level. Some techniques to prevent insider attacks include secure transient associations [F. 99], tamper-proof or tamper-resistant nodes etc. For example, every time a user wants to send a RREQ, the node may require that a user re-key a password or present a fingerprint for biometric analysis to prove their identity. If the device is lost or captured by an unauthorized user, and an attempt to send RREQs is made, this is detected by the node. The node can then destroy its keys to avoid capture (tamper proofing).

3.3.2 Information in Transit

In this subsection, we examine specific threats to **information in transit**. In addition to exploiting vulnerabilities related to the protection and enforcement of the trust levels, compromised or enemy nodes can utilize the information carried in the packets to launch attacks. These attacks can lead to corruption of information, disclosure of sensitive information, theft of legitimate service from other protocol entities, or denial of network service to protocol entities [J. 97]. Threats to information in transit include [WVW, J. 97, W. 95]:

- Interruption: The flow of routing protocol packets, especially route discovery messages and updates, can be interrupted or blocked by malicious nodes. Attackers can selectively filter control messages and updates and force the routing protocol to behave incorrectly. In SAR, a malicious node that interrupts the flow of packets belonging to a higher or lower trust level cannot cause an attack, because it cannot participate in forwarding these packets. If a node filters packets that belong to the same trust level as itself, the broadcast nature of the communication channel can help in detection of interruption attacks by other listeners within transmission range [S. 00].

- Interception and Subversion: Data traffic and control messages, e.g., the "keep-alive" and "are-you-up?" messages can be deflected or rerouted. In SAR, these messages are protected by digital signatures. In addition, the use of flooding makes these attacks superfluous.

- Modification: The integrity of the information packets can be compromised by modifying the packets themselves. SAR provides a suite of cryptographic techniques that can be incorporated on a need-to-use basis to prevent modification. These include digital signatures and

encryption.

- Fabrication: False route and metric information can be inserted into legitimate protocol packets by malicious insider nodes. In such a situation, the sender of the RREQ may receive multiple RREPs. Currently, SAR picks the first RREP that arrives at the sender. The sender can be modified to verify that the RREP has credentials that guarantee the integrity of the metrics and repudiate the ownership of attributes by challenging the intermediate nodes. We plan to incorporate this behavior in the future.

vulnerabilities and passive attacks. Routing updates that reflect transient topology changes can be stored and retransmitted at a later point of time to trigger false updates and false route propagations. SAR provide automatic replay protection by using sequence numbers and timestamps. Most of the attacks described in this section are also called active attacks as the adversaries actively attempt to change the behavior of the protocol. The complement of these attacks are passive attacks, where the behavior of the adversary is more subtle. Examples of passive attacks include covert channels, traffic analysis, sniffing to compromise keys etc. The information inadvertently disclosed to passive attackers by the protocol packets, can be used to launch active attacks. Protection against eavesdropping or sniffing at the MAC layer can be accomplished by using a suitable MAC layer encryption protocol. Protection against passive attacks are difficult in general and many techniques have been proposed to tackle these problems.

3.4 Implementation

In this section, we describe an implementation of SAR built as an augmentation to the AODV [C. 99] protocol in the ns2 [ns2] network simulator. We retain most of ns2 AODV's original behavior, such as on-demand route discovery using flooding, reverse path maintenance in intermediate nodes, and forward path setup via RREP messages. We modify the RREQ (Route REQuest) and the RREP(Route REPly) packet formats to carry additional security information. We call our modified AODV protocol SAODV (Security-aware AODV).

3.4.1 Changes to RREQ

SAODV provides support to enforce the trust hierarchy and also enables customizable security attributes for participating nodes. Three new fields are added into the original AODV RREQ packet format. The first field RQ_SEC_REQUIREMENT is set by the sender and indicates the desired level of trust within an explicit hierarchy for the route to the destination. What values to assign to this field is left to the end users. This field can be used to carry simple integer values reflecting the existing hierarchies in a user's organization. For example, if the application is a military situation, the security requirement field can carry the information about the minimum rank required to relay this communication. In this case, we can use simple integer values to indicate the ranks.

The second field RQ_SEC_QOP_VECTOR contains a bit vector that represents the list of required security capabilities of participating nodes. For example, if the sender chooses to require simple hash, digital signature, and content encryption over the SAODV packets, the matching bits in the field is set to indicate the sender's requirements. The meanings of each bit in the bit vector is left to the deployment.

The last field added to the RREQ packet is the security guarantee, or RQ_SEC_GUARANTEE. This field indicates the maximum level of security afforded by all discovered route. It is updated at every hop during the route discovery phase. If RQ_SEC_REQUIREMENT is represented in integers, RQ_SEC_GUARANTEE is the minimum of the security levels of the participating nodes. If RQ_SEC_REQUIREMENT is represented in bit vectors, RQ_SEC_GUARANTEE is the result of bit-wise AND operations of all the bit vectors representing the capabilities of the participating nodes. This information is copied into RREP and sent back to the sender indicating the actual security the sender can use. The sender can use this security guarantee value to determine whether it needs a more secure connection or not. In addition, SAODV also has support for digital signatures. If the application requested integrity support, a new field to store the computed digital signatures is added to the RREQ.

3.4.2 Changes to RREP

One additional field is also added to the RREP. When an RREQ successfully traverses the network from the sender to the destination, the value of the RQ_SEC_GUARANTEE field in the RREQ packet is copied into the RP_SEC_GUARANTEE field in the RREP packet. The sender can use this value to determine the security level over the whole route. Also, this value is copied into the routing tables of the nodes in the reverse path, to maintain security information about cached routes.

3.4.3 SAODV Route Discovery

In a SAODV route discovery, the source node set the RQ_SEC_REQUIREMENT field to the required level of trustworthiness and RQ_SEC_QOP_VECTOR fields with required set of security capabilities and broadcast the RREQ packet. When an intermediate node receives a RREQ packet, the protocol first checks if the node can satisfy the security requirement indicated in the packet. If the node is secure enough to participate in the routing and also capable of performing the operations listed in the RQ_SEC_QOP_VECTOR field. The first test on the security level is performed by a simple arithmetic comparison and the second test on the QoP bit vector can be done with a bit-wise comparison operator. SAODV behaves like AODV and the RREQ packet is forwarded to its neighbors. If the intermediate node cannot satisfy the security requirement, the RREQ packet is dropped and not forwarded. When an intermediate node decides to forward the request, a new field in the RREQ packet is updated. RQ_SEC_GUARANTEE field is useful in the case where route discovery discovers a route that is more secure than the sender asked for. It is also useful for security-aware applications to get more detailed information about the quality of security for the routes discovered.

This approach opens the question of the effect of malicious nodes in networks. Since it is not uncommon to assume that some mobile nodes will either be captured or compromised during the operation, SAODV must provide a way to guarantee the cooperation of nodes. This cooperation is achieved by encrypting the RREQ headers or by adding digital signatures and distributing keys to nodes that belong to the same level in the trust hierarchy that can decrypt these headers and re-encrypt them when necessary. It is also possible that a mobile node with a set of keys and credentials

is compromised and the keys and the credentials are exposed to adversaries. In such cases, the adversary can assume the identity of the compromised nodes but cannot do anything more than the compromised node was originally allowed to do. Still, it is desirable to design the underlying trust management system to withstand such attacks and localize the effect of compromised nodes.

The arrival of a RREQ packet at the destination indicates the presence of a route from the sender to the receiver that satisfies the security requirement specified by the sender. The destination node sends the RREP packet as in AODV, but with additional information indicating the maximum security available over the route. This information is suitably protected using the same mechanism used to protect the RREQ packet so that only nodes that belong to a particular trust level can process these packets. The value of the RQ_SEC_GUARANTEE field in the RREQ packet is copied to the RP_SEC_GUARANTEE field in the RREP packet. When the RREP packet arrives at an intermediate node in the reverse path, intermediate nodes that are allowed to participate update their routing tables as in AODV and also record the new RP_SEC_GUARANTEE value. This value indicates the maximum security available on the cached forward path. When a trusted intermediate node answers a RREQ query using cached information, this value is compared to the security requirement in the RREQ packet and only when the forward path can guarantee enough security is the cached route information sent back in the RREP.

3.5 Performance Evaluation

In this section, we present the performance evaluation of SAODV protocol. The main goal of our evaluation is to measure the network performance of SAODV using simulation time and delivery ratio of the data packets and also the newly added overhead caused by the SAR-related operations including security-aware route discovery, digital signatures, and encrypted packets. The original AODV protocol is used as a benchmark to study the pure processing overheads of SAODV. Since SAODV enables applications to specify security attributes in their routes, the behavior of SAODV and AODV cannot be compared directly. As our baseline, we use SAODV/Disabled, a protocol that behaves like AODV with respect to its dropping and forwarding behavior, and also includes the additional overheads and modified packet formats of SAODV. The simulations were run for different security attributes, packet formats, traffic patterns, and trust hierarchies. Across our experiments,

we observe that SAODV sends fewer routing protocol control messages (RREPs, RREQs, etc.) for the same number of flows and the same amount of application data. As a result, although the overhead per control message is higher in SAODV, the performance impact is sustainable. We use ns-2 network simulator for our evaluations.

3.5.1 Simulation Set-up

Our results are based on the simulation set up for 50 nodes moving around in a 670m by 670m region. Nodes move according to the random way-point model described in [BMJ+98]. We classify the 50 nodes in our simulations into three levels viz., high, medium and low, each with 15, 15, and 20 nodes respectively. When a node sends out the route request, it uses its own security level as the security requirement for the route. In all of our measurements, we send the same amount of data (about 10000 packets) for the same number of flows (20) at the same rate. Our simulation is run until the flows complete sending all packets. Within this set-up, we measure and compare the overall completion time and the number of control messages sent. Two different traffic patterns are used to drive the simulations. Traffic pattern 1 consists of 20 CBR flows. 10% of the flows are between the high level nodes, 20% between the medium and 70% between the low level nodes. Traffic pattern 2 also has 20 CBR flows, but the distribution is 33%, 33%, 34% for the high, medium, and low level nodes. Pattern 1 will generate three levels of overlays where low-security layer is heavily populated with and the medium- and high-security layers are relatively sparse. In comparison, Pattern 2 generates three levels of overlays where high-security layer contains about 33% of mobile nodes, medium-security layer 66%, and high-security layer 33%, resulting in better-connected overlays at the higher-security layers. The packet size is 512 bytes, and the sending rate is 4 packets/second. The maximum number of packets in each flow is 500. In Section 3.5.2 we measure the overhead for enforcing the trust hierarchy. Section 3.5.3 presents our results for secure routing, specifically, the overhead of adding encryption and digital signatures to SAODV's RREPs.

3.5.2 SAODV Processing Overheads

SAODV has larger RREQ and RREP packets compared to AODV and all nodes participating in route discovery must do additional processing to check the security requirements and update the

security guarantee. Initially, SAODV is configured to do the trust enforcement processing, but not drop the RREQ packets when it is supposed to. As mentioned earlier, this version is a baseline in performance measurement to show how SAODV processing affects protocol behavior. The pure overhead of the SAODV modification is measured in comparison to the original AODV.

Table 3.1: Overall Simulation Time

	Traffic Pattern 1	**Traffic Pattern 2**
AODV	2803	2844
SAODV/Disabled	2844	2918

As shown in Table 3.1, with traffic pattern 1, SAODV takes 1% longer to finish the whole simulation and with traffic pattern 2, less than 3% more. Pattern 2 includes more flows with higher security requirements that requires longer processing time for packets. This means that the pure overhead of adding additional processing to enable security, in the absence of dropping, is not prohibitive. We use this SAODV without RREQ dropping, SAODV/Disabled, as our baseline for rest of the performance measurements.

Route Discovery

Next, we ran SAODV/Disabled and SAODV with explicit trust values on the same traffic patterns to observe the difference in protocol behavior. The number of routes discovered by SAODV/Disabled and SAODV and the number of routes that violate the security requirements in SAODV/Disabled were recorded. Since SAODV/Disabled behaves like the original AODV, some of the routes found violated the security requirements. This is summarized in Table 3.2.

Table 3.2: Number of Routes Discovered

	Traffic Pattern 1	**Traffic Pattern 2**
Total number of route discovery by SAODV/Disabled	93	95
Routes violating security requirement by SAODV/Disabled	14	19
Routes discovery by SAODV	80	73

Although SAODV/Disabled found more routes when the trust levels were enforced, 14 and 19 of these routes respectively were unusable. SAODV discovered fewer routes, but these routes are guaranteed to obey the trust requirements of their senders.

Routing Message Overheads

Table 3.3 shows the numbers of routing protocol messages in SAODV/Disabled and SAODV. We observe that there is a drop in the number of RREQ messages sent in SAODV. This is because a RREQ is dropped and not forwarded when the intermediate nodes cannot handle the security requirements of the RREQ packets.

Table 3.3: Routing Message Overhead

	RREQ		RREP		Routing Msgs	
	Pattern 1	Pattern 2	Pattern 1	Pattern 2	Pattern 1	Pattern 2
SAODV/Disabled	2333	2566	107	102	2410	2668
SAODV	2285	1504	80	73	2365	1577

These numbers imply that SAODV generates fewer routing messages, while enabling applications to find more relevant routes. In the case of Pattern 1, there was a decrease of 2% in RREQ messages and 25% in RREP messages. For Pattern 2, the results were more accentuated (41% in RREQs, and 27% in RREPs). The reason is that the trust hierarchy is more equitably distributed in Pattern 2 and routes tend to be shorter.

Overall Simulation Time and Transmitted Data

SAODV restricts the route discovery process to only routes that can satisfy the requirements. This feature may force packets to follow longer but more secure routes and result in taking more time to finish the communication. The overhead of the protocol is illustrated in Table 3.4. The overall time to complete transmission of all of the traffic flows in both SAODV with trust enforcement and SAODV/Disabled and the total amount of data transmitted are illustrated in the Table 3.4.

Table 3.4: Overall Simulation Time and Transmitted Data

	Simulation Time		Transmitted Data	
	Pattern 1	Pattern 2	Pattern 1	Pattern 2
SAODV/Disabled	2844	2918	10023	10022
SAODV	2911	2925	10028	10017

With RREQ dropping, SAODV takes 2.3% and 0.2% more time to finish in traffic patterns 1 and 2 compared to SAODV/Disabled. Patter 1 includes a small fraction of higher level nodes there

any traffic originating from such higher level nodes goes through much longer routes. Although SAODV takes marginally more time to finish communication, it still finds routes in most cases, and delivers almost the same amount of data from senders to the receivers as shown in the table.

Route Optimality

The data packets may follow longer routes in SAODV if the shorter routes cannot satisfy the security requirements. Table 3.5 shows the length of the routes each data packet travels compared to the shortest possible routes at the time. The optimal length means that the packet actually follows the shortest route between the sender and the destination at the time of transmission. The table lists the the number of packets that traversed a route with each length. With Pattern 1, we see that a significant number of packets take longer routes, especially among transmissions in the higher trust levels (34%). The impact is not as severe in traffic pattern 2 (14%) since the trust hierarchy is more equitably distributed in pattern 2.

Table 3.5: Route Optimality

	Optimal Length		Optimal Length + 1		Optimal Length + 2		Optimal Length + 3	
Traffic Pattern	1	2	1	2	1	2	1	2
SAODV/Disabled	7700	8570	2142	1266	130	132	0	4
SAODV	6481	8549	2414	1273	1075	130	0	3

3.5.3 Secure Routing Measurements

The SAODV protocol can be augmented with hash digests (SAODV/Digital Signature) and symmetric encryption mechanisms (SAODV/Encryption). The overhead of including security attributes in the RREQ messages are presented in this subsection. The signed hash digests provide message integrity, whereas encrypting packets guarantees their confidentiality. Nodes that have the same trust level share the same encryption and decryption keys. The MD5 Hash algorithm [Riv92] and the Blowfish block cipher [Sch94] were used for these measurements. We present the measurements for Traffic Pattern 1 only. The results for Pattern 2 show a similar trend.

The entire RREQ packet was encrypted, with the exception of the packet-type field. For SAODV with digital signatures, an additional field was added to the RREQ header. The MD5

Table 3.6: Routing Message Overheads for Secure Routing

	RREQ		RREP		Routing Msgs	
	Encryption	Signed Hash	Encryption	Signed Hash	Encryption	Signed Hash
SAODV/Disabled	2225	2219	77	85	2378	2381
SAODV	2175	2148	74	80	2341	2311

hash algorithm was used to generate a MAC (Message Authentication Code), along with Blowfish encryption to protect the integrity of the MAC. The SAODV/Disabled protocol reflects the overhead of adding the extra field in the header. In Table 3.6, we observe that SAODV/Encryption and SAODV/Digital Signature sent fewer RREQs and RREPs than SAODV/Disabled. This is because nodes that were not capable of decrypting the encrypted RREQ packets, or could not verify the signatures, dropped these packets without forwarding. SAODV/Encryption showed a 9.1% decrease and SAODV/Digital Signatures showed a 17% decrease. This reinforces our claim that SAODV sends fewer control messages (RREQs and RREPs) than SAODV/Disabled, although each packet needs more processing.

Table 3.7: Overall Simulation Time and Transmitted Data

	Simulation Time		Transmitted Data	
	Encryption	Signed Hash	Encryption	Signed Hash
SAODV/Disabled	2899	2875	10024	10025
SAODV	2918	2933	10026	10017

Table 3.7 presents overall simulation time and transmitted data. Adding encryption increases overall simulation time by 0.7% and adding digital signatures by 2% (in addition to the trust enforcement overheads). However, the number of packets transmitted was approximately equal.

3.6 Summary of Contributions

SAR enables the discovery of trustworthy routes in a mobile ad hoc environment. Its integrated security metrics allow applications to capture and enforce explicit cooperative trust relationships instead of blindly trusting the intermediate nodes to be trustworthy. In other words, nodes in SAR can *measure* the trustworthiness of the intermediate nodes based on the available situational

information therefore of the route. Measurement of trustworthiness is performed for each route discovery process, enabling constant monitoring of the changing situation in an ad hoc network. The design of SAR can be easily incorporated into generic ad hoc routing protocols as illustrated by our implementation example - SAODV. The processing overheads in SAR are offset by restricting the scope of the flooding for more relevant routes, providing comparable price/performance benefits.

SAR is not a stand-alone security service. It requires the support for secure routing that secures the control messages along with a reliable authentication service for identifying all participants. In the next four chapters, we present our work in situation-aware authentication service provided by ad hoc key management frameworks.

Chapter 4

Situation-Aware Key Management for Ad Hoc Networks

Security begins with reliable authentication. Most fundamental security services, including access control, auditing, and authorization rely on an authentication service to provide reliable identification of participants. A well-established way to provide an authentication service in distributed systems is public key cryptography [W. 76], where an entity is represented with a pair of keys. The public key is used as the ID of the entity while the private key is used to prove the ownership of the public key. The public key is disseminated in the form of a digital certificate that binds the entity's identity to the entity's public key. The successful use of public key cryptography requires an efficient mechanism to manage such digital certificates. In this chapter, we investigate the challenges of providing a key management service within the limits of ad hoc networks and demonstrate the need for situation-awareness in ad hoc key management.

4.1 Key Management in Ad Hoc Networks

Various designs for ad hoc key management frameworks have been proposed. However, it is important to note that none of these approaches have been shown to provide a universal solution in dynamic environments. This limitation mainly comes from the fact that most approaches try to adapt solutions from wired environments without adequately addressing the specific challenges in ad hoc networks. In other words, *hidden assumptions* were not clearly identified during the design phase. To understand why previous approaches fail, we first identify three fundamental goals of ad hoc key management. Then, we follow up with two underlying principles for key management in ad

hoc networks to expose the underlying hidden assumptions in ad hoc key management framework designs: *node participation* and *the use of a trusted third party*. While most previous approaches adhere to one of these principles, it is often at the expense of the other, resulting in limited success. After defining each principle in detail, we discuss existing approaches with an emphasis on how they achieve or fail to achieve these proposed principles. To capture the changing situation into a intuitive metric for end users, we present our trust model to capture trust relationships among the mobile nodes in the network and our *Metrics of Authentication (MoA)* that translates the trust relationships into a concise and intuitive measure. Finally, we present the conceptual design of a situation-aware key management framework that we will use as the design guideline in the following chapters.

4.1.1 Three Goals for Ad Hoc Key Management

A successful key management framework for ad hoc networks must satisfy three fundamental requirements: *fault tolerance*, *security*, and *availability*. These terms are sometimes used interchangeably, mainly because they are not independent of each other. To avoid confusion, we first clearly define these terms.

- *Fault Tolerance*: The goal of fault tolerance is to maintain correct operation in the presence of faulty nodes. We restrict the definition of faulty to non-malicious. Fault tolerance is related to availability because as long as the service can tolerate the faults, such faults do not impact service availability.

- *Security*: Acting as the trust anchor for the whole network, a key management framework must be designed to be resilient against all levels of attacks and robust enough to withstand a relatively high fraction of compromised nodes.

- *Availability*: Traditionally, the term availability has been used in conjunction with fault tolerance. However, in ad hoc networks, availability is also highly dependent on network connectivity which, in turn, is affected by the node mobility. In wired networks, if the service is online, it is by definition available since connectivity between clients and the service is usually guaranteed. In ad hoc networks, clients may not be able to contact an operational

service due to unstable and rapidly changing connectivity.

These three requirements are not independent of each other. For that reason, careless attempts to improve any one aspect may affect the others adversely. For example, a simple and very effective way to improve fault tolerance of PKI is to replicate the CA. However, this approach makes the overall framework more vulnerable. Interactions between these requirements must be throughly understood before designing a distributed key management framework.

To understand these interactions in ad hoc environments, we present two design principles for ad hoc key management in the next section. These principles are based on the unique characteristics of ad hoc environments and used to guide the design of ad hoc key management frameworks that operate within the limitation of ad hoc environments.

4.1.2 Two Principles for Ad Hoc Key Management

In this section, we discuss the details of two key principles for ad hoc key management framework design. Most current key management frameworks for ad hoc networks implicitly embody one of the two underlying principles, while sacrificing or ignoring the other.

Node Participation

The *node participation* principle states that a key management framework for ad hoc networks should rely on a large number of nodes for availability, but a smaller group of nodes for security. This is to maintain a balance between two potentially conflicting goals of fault tolerance and availability. Given the physical vulnerability of mobile nodes in ad hoc networks, it is not effective to burden a single node with the responsibility of providing a security service like key management. A natural way to address this problem is to distribute the security service over multiple nodes, hence improving fault tolerance. In general, this group can span from a single node to all nodes in the network. However, blind and equal distribution of security functionality over too many nodes leads to a vulnerable system. This observation leads to two important questions as to the participation of nodes in key management. First, *How many of the nodes should participate?* The participation of a higher fraction of nodes in the network, can improve availability and fault tolerance. However, without careful consideration, higher participation can also lead to higher vulnerability. This leads

to the second question: *How should the nodes participate?* When a security service is divided across a large number of nodes with equal responsibilities, the availability of the service increases since there are more nodes that an end user can contact. However, this improved availability also helps adversaries locate and compromise these nodes and eventually compromise the security of the service. Therefore, a blind and equal distribution of functionality to multiple nodes can degrade the overall security. Instead, core functionalities of the security service should be distributed to a restricted set of secure nodes, providing strong security and high fault tolerance. The rest of the nodes share lower level functionality to improve the availability of the core nodes. Compromising any of these low level nodes should not compromise overall security, but only affect the availability of the core service.

Use of a Trusted Third Party

The use of a trusted third party (TTP) principle states that a key management framework should use a TTP to maintain a high level of security of the framework to provide high quality authentication services, effectively requiring the use of PKI. Without a clear trust anchor in the network, authentications can only rely on casual trust relationships. Since there are no guarantees about the behavior of participating nodes, any authentication based on such casual relationships cannot be trusted for security-sensitive applications. A TTP provides a trust anchor that can be used as the basis for further trust relationships. Since every node trusts the TTP, authentication provided by the TTP is trusted with a high level of confidence. Since it is not trivial to maintain any form of a TTP in ad hoc networks, it may appear attractive to use a fully decentralized key management framework that does not rely on a TTP. However, in any approach without a TTP, an authentication must rely on casual and voluntary relationships between nodes or an accumulation of such relationships. Essentially, without trustworthy authentication, no further security service can be built to guarantee a high level of assurance. Therefore, using a TTP is crucial for any ad hoc network with strong security requirements. When more than one TTP is available in a single network, they can be used to improve the quality of authentication even further. However, we do not investigate the issue of using more than one TTP(s) in this thesis.

Given these two principles for key management in ad hoc networks, we now discuss current

approaches and how they succeed or fail in supporting these principles.

4.1.3 Certificate Chaining

Authentication by a chain of authorities has commonly been used in large scale dynamic networks without a single authority [Ken93, Zim95]. In general, authentication is represented as a set of digital certificates that form a chain. Certificate chaining does not require heavy infrastructure or complex bootstrapping procedures and every node has identical roles and responsibilities. In certificate chaining, any node can issue a certificate to any other node based on its own discretion. Given the inherent peer-to-peer nature of certificate chaining, it is easy to add new nodes into the system and extend the coverage of the certificate chaining system. Since an authenticating node only utilizes local information and does not rely on an external system, certificate chaining is very tolerant to faults by definition. These characteristics of certificate chaining make it a potential candidate for key management in ad hoc networks, as realized by Hubaux et al. [CBH03]. Certificate chaining achieves the maximum level of node participation, since every node can participates by issuing certificates to each other to populate the certification graph. However, every node shares the same responsibilities, limiting the security of the system. Additionally, certificate chaining fails to use any TTP. This departure from the the principles leads to two main limitations.

First, since participating nodes operate in a best-effort manner, there can be situations when authentication cannot be provided. Essentially, a certification graph may not be populated enough to provide certificate chains for a given pair of nodes. Since there is no means to force mobile nodes to issue certificates and keep the certification graph dense enough, it is difficult to predict if any given authentication request can be fulfilled, impacting the availability goal. As shown in two studies [CBH02, McB98] of PGP [Zim95], a 1998 snapshot of the PGP certification graph that included 57582 nodes only had 3100 nodes (5%) in its largest strongly connected component (SCC) [McB98], while in a more recent snapshot, there is an even larger gap between the total number of nodes and the size of the largest SCC (2.5%) [CBH02]. Essentially, the currently deployed PGP system has one large SCC that contains a very small fraction of the nodes and most of the nodes are scattered to form a sparse graph. This gap is important since only members of the same SCC can authenticate each other. If a node is outside a SCC, two-way authentication between the

outside node and any other node is not guaranteed. Considering that PGP has been in operation for years (since the early 90's), these characteristics are not expected to change in the near future. As Capkun suggested in his study of the small world property of PGP graphs, this characteristic is typical of all PGP-like certificate chaining systems [CBH02]. However, it still remains a question if the same property will hold true in ad hoc environments.

Second, without relying on a TTP, any trust relationships must rely on the goodwill and the correct behavior of all participants. Any single misbehaving or malicious node participating in a certificate chain can taint the whole chain and invalidate the authentication. However, since there is no clear way to tell if a certificate chain includes any misbehaving nodes, the overall confidence value of certificate chains must be relatively low. To combat this problem, several enhancements have been proposed, including limiting the chain length and using multiple node-disjoint chains [RS99] and detecting discrepancies from multiple sets of chains [JRN]. Despite these improvements, the quality of authentication provided by certificate chaining may still not be high enough to support strong security goal, and it must be quantified to provide any meaningful authentication.

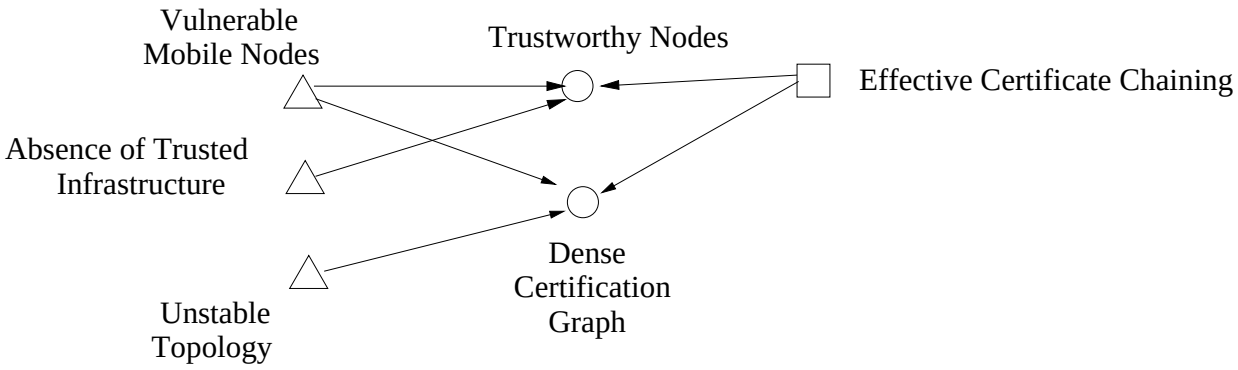


Figure 4.1: Situation Diagram for Certificate Chaining

Quality of authentication via certificate chaining depends on two network properties: the trustworthiness of the participating nodes and a high enough participation ratio of mobile nodes. The trustworthiness of mobile nodes is adversely affected by two underlying environmental factors: the inherent vulnerability of mobile nodes and also by the absence of a trust infrastructure. In other words, mobile nodes that participate in certificate chaining cannot be completely trusted and any authentication via certificate chaining must be discounted for that reason. The effectiveness of certificate chaining is also affected by the amount of participation of mobile nodes. Since the formation

of a certification graph is left to voluntary cooperation of mobile nodes, it cannot be guaranteed that the resulting certification graph will contain enough certificates to be useful. Therefore, the formation of a certification graph is again adversely affected by the vulnerability of mobile nodes and also unstable network topology. Based on these observations, we identify the situation diagram for a certificate chaining system as represented as in Figure 4.1.3.

4.1.4 Distributed CA Approaches

To address the unique challenges in ad hoc networks, several distributed CA approaches employ threshold cryptography to securely distribute the CA's functionality over multiple nodes [KZL+01, WMB99, YK03, ZH99]. In this chapter, we focus on how each of these approaches satisfies or does not satisfy the design principles. CA functionality is distributed in such a manner that an adversary must compromise a certain fraction of the key shares to compromise the distributed CA itself. At the same time, an end user need only access a subset of the distributed CA nodes to get certification services. Wu et al. first suggest a distributed CA based on threshold cryptography [WMB99] and Zhou et al. propose its application to ad hoc networks [ZH99]. Kong et al. [KZL+01] follow through by designing full key management frameworks followed by our own MOCA framework [YK04b]. It is clear that all distributed CA approaches employ the use of a TTP principle, satisfying the security goal of ad hoc key management. While all distributed CA approaches appear to similarly address the node participation principle, the approaches differ in *how* they choose nodes to participate and this difference has impacts on all three goals: fault tolerance, security and availability. Kong et al. proposed a distributed CA solution where every mobile node in an ad hoc network acts as a CA node and shares the responsibility of a CA [KZL+01]. This approach maximizes node participation by utilizing all nodes in the network, achieving very high availability. However, their solution is vulnerable to adversaries that can compromise a relatively small number of mobile nodes, and also to Sybil attacks [Dou00]. Essentially, this approach violates the security component of the node participation principle by involving all nodes in the core security function and fails to meet the security goal of ad hoc key management. In response to these limitation, we designed MOCA, a generalized key management framework for all possible configurations of distributed CA approaches based on threshold cryptography (See Chapter 5 for details). The results from MOCA suggests

that the fraction of CA nodes should be kept relatively small to maintain strong security. This fits well with the security component of the node participation principle that limits the main key management functions to a small fraction of nodes. However, the MOCA framework sacrifices the first part of the principle, and so availability, by not involving the rest of the nodes in any part of key management. Essentially, Kong et al.'s approach sacrifices the security to achieve ubiquitous availability while MOCA sacrifices availability to maintain strong security.

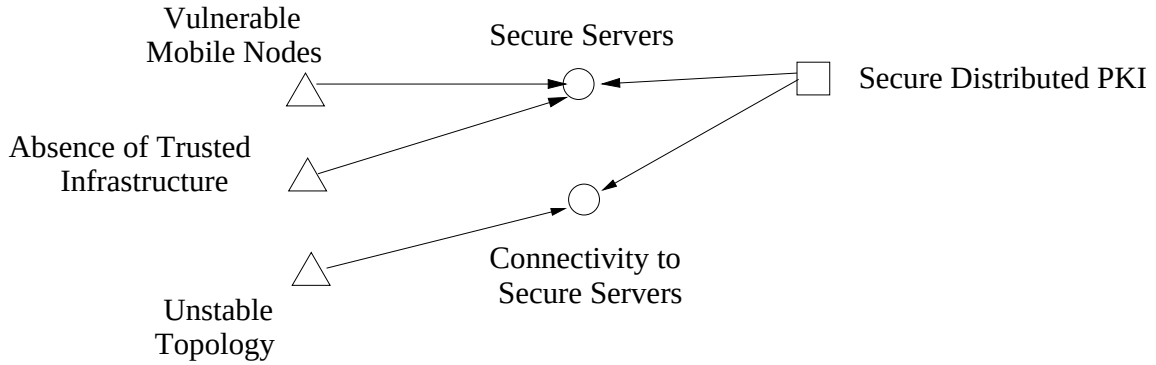


Figure 4.2: Situation for Ad Hoc PKI

To provide a secure distributed PKI, two network properties are required: availability of secure and reliable server(s) and stable connectivity between such servers and the rest of the network. However, maintaining a secure and reliable server inside an ad hoc network is not a trivial task due to the underlying environmental factors such as inherent vulnerability of mobile nodes and the absence of any reliable infrastructure. Also, the connectivity between the secure servers and the rest of the network cannot be guaranteed because of the unstable network topology caused by the mobility of nodes. These requirements are captured in the situation diagram in Figure 4.2 and we later use this diagram to drive the design of the MOCA framework as well as the metrics to measure the quality of authentication provided by distributed PKIs.

Figure 4.3 presents a visual comparison of existing distributed PKI approaches. The ideal goal of ad hoc PKI is displayed as the solid black rectangle at the upper-right corner where both QoA and availability remains high and show minimal fluctuations. In comparison, a single CA scenario shows a low QoA since it is not very difficult to compromise the single CA. Availability of the single CA is also low due to the changing topology and node mobility. When the CAs are replicated, the availability is improved at the cost of degrading the QoA even further since the attacks on a

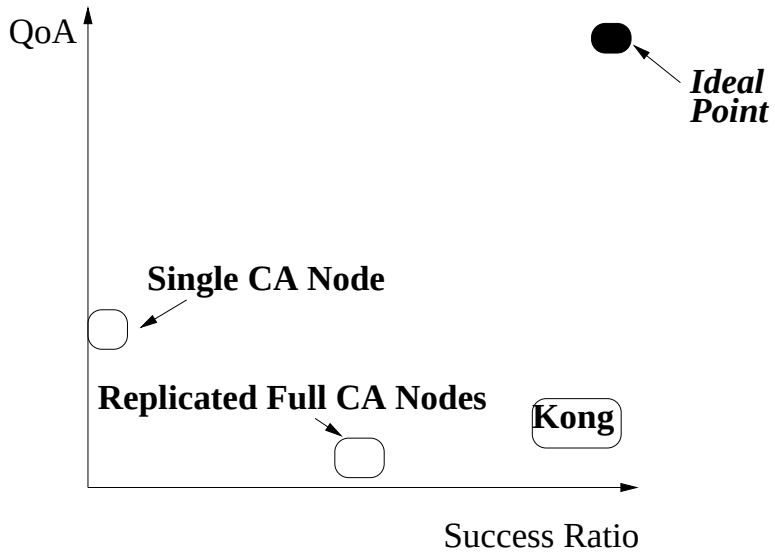


Figure 4.3: Comparison of Existing Ad Hoc PKIs

CA nodes become even easier. Kong's approach shows an extreme case where maximal availability sacrifices the security of the framework, resulting in a very low QoA. In the next section, we present the conceptual design of our situation-aware key management framework based on the observations presented so far.

4.2 Situation-Aware Key Management Frameworks

As shown in the previous sections, none of the existing ad hoc key management frameworks provides a perfect and ideal authentication service due to the challenging environmental effects of ad hoc environments. The goal of a situation-aware key management service is to provide basic authentication services augmented by a measure of the *Quality of Authentication (QoA)* so that end users can accurately understand the quality of the provided authentication service and act accordingly. QoA for key management services can be measured based on the trust relationships in the system and observations about network conditions. All of these components are affected by the current *situation*. Since an end user cannot be expected to understand the details of the key management framework designs or the constantly changing conditions of the network, a situation-aware key management service must provide an interpretation of all of this *situational* information into a simple and intuitive measure of QoA. The design of such a framework is captured in Figure 4.4.

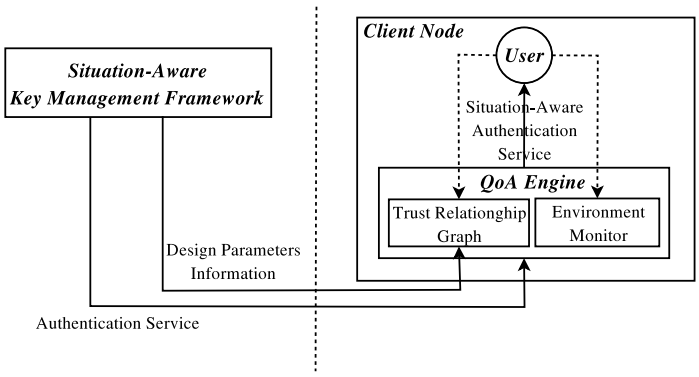


Figure 4.4: Situation-Aware Key Management Framework

Each client node in the ad hoc network includes an *end user* and a situation-aware software agent called the *QoA Engine*. The responsibility of the *QoA Engine* is to take input from the basic key management service, the end user, and observations about network conditions and provide the end user with an augmented authentication service with the QoA information, forming the *situation-aware authentication service*. Based on this QoA, the end user can either decide to use the provided service or reject it.

The output QoA is calculated using well-defined *Metrics of Authentication (MoA)* that use the situational information captured by the two components of the QoA Engine: the *Trust Relationship Graph (TRG)* and the *Environment Monitor (EM)* as seen in Figure 4.4. The TRG keeps track of the trust relationships from the end user's view, using inputs from both the key management service and the end user. The EM monitors the changes in relevant network conditions, such as the vulnerability of mobile nodes, the impact from the absence of a trusted entity, and the effects of network instability. These environmental factors are collected from multiple sources including the end user, the key management framework and the EM's own monitoring function. The detailed effects of each situational factor on the final QoA are specific to the design of the key management framework and we will discuss in detail how our MoA reflects these different effects into QoA measurements in Section 4.2.2.

4.2.1 Modeling Trust Relationships

All authentication services for a distributed system are based on the trust relationships among the entities in the system and the shape of these trust relationships is heavily affected by the changing

situation. Since QoA is the means to integrate and express the changing situation, measuring the QoA of an authentication service must begin with modeling these trust relationships. A digital certificate is a popular way of making trust relationships explicit. When one entity has a certain amount of trust in another, the first entity can issue a digital certificate to that effect.

A natural choice of a data structure for representing and capturing trust relationships is a form of directed graph. Entities in a distributed system can be modeled as vertices and trust relationships among them can be captured as edges. Based on a large body of previous work in certificate chaining models [BBK94, BLNS86, Ken93, LA98, RS99, Zim95] and some more relevant approaches for ad hoc environments [YK04a, YK04b], we adopt a weighted directional graph structure called a *Trust Relationship Graph (TRG)*, $TRG = \{V, E\}$, where V is the set of vertices representing the mobile nodes in the system and E is the set of edges representing the trust relationships among the mobile nodes. Each edge $e_i \in E$ has a weight function $c(e_i) = c_i$ that expresses a confidence value, which captures the amount of trust that the issuer of the certificate has on the target entity. To avoid limiting the applicability of the model, we allow any confidence value between 0.0 (no trust) and 1.0 (absolute trust). The specific value assigned by a user to an issued certificate is determined by the user for a given network deployment. Network operators can issue guidelines on appropriate methods to determine confidence values. The Thawte web of trust system enforces two entities to meet in person to verify identities before any certificate can be issued [Tha]. In general, a confidence value is calculated based on the number and quality of identification materials used to prove the target entity's identity. Al-Muhtadi proposed a novel way to incorporate inputs from many authentication mechanisms into a single representative value using fuzzy logic in his PhD thesis [AM05]. His design is essentially a formal model of combining different authentication evidences as used in the Thatwe system. Similarly, interactive trust building systems like TrustBuilder [Win03] can be employed to calculate appropriate confidence values between any pair of entities. Another possible approach is to allow each end user to actively communicate with their peers to collect as much information as possible. However, this approach comes at a high communication cost, which usually cannot be afforded in ad hoc environments. Therefore, we only focus on passive information gathering and the best ways to reason about collected information throughout this thesis. Figure 4.5 illustrates the simplest form of a TRG that contains only two

Figure 4.5: A Simple Trust Relationship Graph

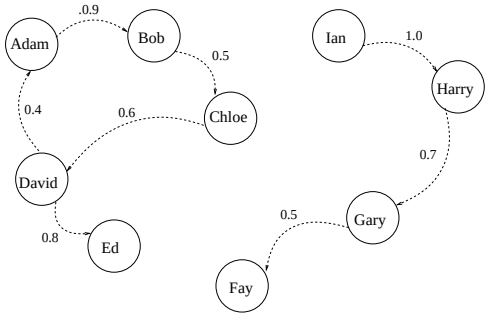


Figure 4.6: Trust Relationship Graph of a Certificate Chaining System

vertices and one edge. This represents a network of two mobile nodes, Alice and Bob, with one certificate that Bob issued to Alice. The confidence value 0.8 is assigned by Bob showing the level of trust Bob has in Alice. A more complex TRG from a typical certificate chaining system looks like Figure 4.6 where many certificates populate the TRG. It is important to note that each client node in a network may have a differently shaped local TRG based on their observations and knowledge, which intuitively maps to different end users having different views about other users. As a client node collects more information and the local TRG contains more information, the end user can put higher confidence on the authentication result. We discuss this issue in detail in Section 4.2.2.

We propose the idea of extending this trust graph model to include trusted third parties, such as certificate authorities [YK04b]. Trust graphs for such systems generally look like Figure 4.7. For distributed CA approaches, the CA node in the graph represents the collection of nodes that comprises the distributed CA. Every node in the network trusts the CA as the trust anchor for the whole network. Since the confidence in authentication from a CA-based system solely depends on the security and reliability of the CA, users should be able to express their own perception about the security and reliability of the CA.

In the remainder of this section, we elaborate further on the CA security level metric. We define the CA's security level as the probability that an adversary can compromise the CA. This probability can be calculated from the configuration parameters of the distributed CA, including the total number of nodes in the network, M, the number of CA nodes, n, and the crypto threshold,

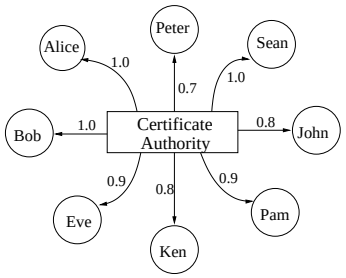


Figure 4.7: Trust Relationship Graph of a PKI

k ($1 \leq k \leq n \leq M$). In addition, each end user must provide the final parameter, c, the capacity of the attacker(s), which represents a simple threat model: there is one or more active and colluding attackers at a given point of time and they are capable of compromising at most c mobile nodes. If there are multiple independent attackers that do not cooperate with each other, each attacker must be given different c values depending on their capability. Each end user must estimate c based on their perception of network conditions and the behavior of the distributed CA. If $c < k$, the threshold cryptography protects the distributed CA from being compromised. However, if $c \geq k$, the adversaries could compromise enough nodes to break the threshold cryptography. If the adversaries are capable of pinpointing the attacks on only the CA nodes, the framework is always compromised as long as $c \geq k$. However, if the adversaries does not know the locations or the identities of the n CA nodes, the best approach is to compromise as many nodes as possible, hoping that it will allow them to compromise at least k CA nodes, therefore compromising the distributed CA. Given these assumptions, the security level of a distributed CA can be calculated as:

$$\text{CA Security Level} = 1.0 - \frac{\sum_{i=k}^{c} \binom{n}{i}\binom{M-n}{c-i}}{\binom{M}{c}}. \tag{4.1}$$

The second component of this equation captures the probability that a distributed CA is compromised by an attacker with an attack capacity c. The denominator, $\binom{M}{c}$, is the number of all possible cases where the attacker compromises any c nodes in a M-node network. The numerator counts the number of cases where the attacker successfully compromises the distributed CA by compromising k or more CA nodes ($\binom{n}{i}$) and the rest from non-CA nodes ($\binom{M-n}{c-i}$). Hence, this second component represents the probability that an attacker can compromise a distributed PKI by

compromising c mobile nodes. Therefore, Equation 4.1 captures the probability that the attacker *fails* to compromise the CA or the CA resists the attack, and so represents the CA's security level.

Next, we present our *Situation-Aware Metrics of Authentication* that are used to calculate the QoA based on the TRG model presented in this section and the relevant situational information.

4.2.2 Metrics of Authentication

Quality of Authentication (QoA) can be loosely defined as the level of confidence that can be put on an instance of authentication based on the current knowledge about the *situation* of the authentication service. Once the trust relationships among the nodes are captured, the QoA of each authentication instance can be calculated using *Metrics of Authentication (MoA)* that capture the effects of the current situational parameters. Given a trust relationship graph of the network, TRG, the authenticating node, a, and the target node, t, a function $QoA(TRG, a, t)$ returns a numeric value that represents the amount of trust the authenticating node has in the authentication of the target node. The design of our MoA is based on a large body of previous research in authentication metrics [BBK94, BLNS86, Jos99, Ken93, LA98, Mau96, MT03, RS99, TH92, Zim95]. Our contributions include (1) a novel way to measure the security level of distributed PKI approaches, (2) a clean and intuitive method to combine multiple observations in an authentication process, and (3) integration of situational information into the QoA measurement. In the remainder of this section, we first briefly compare two different types of authentication supported in our MoA and then describe how to evaluate the QoA in detail.

Direct vs. Indirect Authentication

Since authentication can come from many different sources, it is important to distinguish between two types of authentication: *direct* and *indirect*. A *direct* authentication is achieved when the authenticating node or the trusted third party has a direct trust relationship with the target node (i.e., the target node possesses a certificate issued by either the authenticating node or the trusted third party). An *indirect* authentication is achieved via a chain of certificates that originates at the authenticating node or the trusted third party and ends at the target node. Indirect authentication relies on the concept of *transitive trust* and is inherently less trustworthy than direct authentication

due to its reliance on other nodes that the authenticating node does not have a direct trust relationship with. Therefore, it is intuitive to favor trust from direct authentication over any indirect authentication. QoA for a direct authentication is defined as the confidence value of the certificate issued to the target:

$$QoA(TRG, a, t) = c(e_i), \text{where } e_i = (a, t), e_i \in E.$$

When it is not possible to directly authenticate the target, the authenticating node bases its decision on a combination of all possible indirect authentications from the TRG.

Calculating the Chain Confidence Value for a Single Chain

Given a certificate chain, there is a single *Chain Confidence Value (CCV)* that is calculated from the edge confidence values and the length of the chain. First, edge confidence values of all of the edges in the chain are multiplied. Then, an attenuation factor is multiplied to discourage the use of longer chains. Let e_i be the edges in the chain c_i, $c(e_i)$ be the confidence value of e_i, p be the probability of a node being compromised, and d be the length of the chain. The CCV is calculated as follows:

$$CCV(c_i) = \prod_{e_i} c(e_i) * (1-p)^{(d-1)}. \tag{4.2}$$

Considering that the validity of a certificate chain relies on the correctness of every participating node, it is intuitive to see that a long certificate chain is more vulnerable and its use should be discouraged if a shorter alternate is available. If we assume that each node in the network is equally likely to be malicious or be compromised with a probability p, the probability that a chain of length d is intact can be denoted as $(1-p)^{(d-1)}$ (not $(1-p)^d$ since the first hop is from a trusted node). To accommodate this observation, the result from the multiplication of edge confidence values is again multiplied by the attenuation factor, $(1-p)^{(d-1)}$. This attenuation factor decreases exponentially as the chain length grows, which effectively discourages the use of long chains. The use of this user-tunable parameter p is a case of using situation-aware information in QoA measurement where the end user's perception of the network condition (i.e., the fraction of compromised nodes) affects the final QoA value. An end user can exercise various methods to collect the information about network

conditions to determine the appropriate p value. If the network is equipped with intrusion detection capability, the monitored network behavior can be provided to end users for their determination of p. An end user can also collect other trusted nodes' opinions on network status and aggregate such opinions to form a collective output of the p value. Exact semantics of determining p is left to each instance of network deployment and available support functionalities.

When the discovered chain originates from a trusted third party instead of from the authenticating node, the trusted third party's security level, as calculated in Section 4.2.1, is multiplied to get the final CCV:

$$CCV(c_i) = \prod_{e_i} c(e_i) * (1-p)^{(d-1)} * CA\ Security\ Level. \tag{4.3}$$

Pitfalls of Using a Single Certificate Chain

Other than the intuitive reason that using all available information enables the authenticator to reach a more informed decision, using a single certificate chain for authentication has one serious flaw. Reiter and Stubblebine first pointed out that it is trivial to manipulate the confidence values in a single chain since the attacker can simply issue many bogus certificates [RS99]. Reiter proposed using multiple chains that exist in the authenticator's local TRG with one restriction. Simply accommodating all chains in a TRG is again vulnerable to attacks from a small number of malicious nodes since they can issue any number of bogus certificates and it is trivial to affect many certificate chains. Reiter then suggested the use of node-disjoint chains only. In that way, the impact any one malicious node can have on the final result is limited to a single chain where the particular malicious node is included. As long as there are enough chains in the trust relationship graph, the effect of a small number of malicious nodes can be effectively masked. In our study, we follow Reiter's guideline and also limit the certificate chains to be node-disjoint from one another. Once a set of node-disjoint certificate chains are identified and their individual CCVs calculated, the next question is how to combine these individual CCVs into a single, meaningful yet concise result. We use results from a research area in philosophy that tries to model the human behavior of reasoning. More specifically we use the wisdom from an area called *Scientific Confirmation Theory* [Car50].

If multiple chains exist between two nodes, it is beneficial to leverage information from all chains

to make a more informed decision about the trustworthiness of a particular node, increasing the accuracy of the calculated QoA. Therefore, the final QoA should be a function of the CCVs for the multiple chains. There are a small number of results from the research community on this matter. Reiter and Stubblebine proposed to treat multiple certificate chains as a set of network paths with respective capacity and use the maximum network flow as the quality of authentication result [RS99]. Jiang et al. proposed to use the detection of *conflicting chains*, which are a set of chains with conflicting opinions about a single target, to improve the robustness of the authentication decision [JRN]. While both approaches essentially try to achieve the same goal, the combination of the opinions from multiple certificate chains without getting manipulated easily by adversaries, neither completely achieves the goal. Reiter's approach eventually shifted the burden to an external insurance system where the value of the maximum network flow is matched to a monetary insurance each intermediate node is willing to vouch for. Jiang's approach solves some unanswered questions from the previous results but still does not provide a comprehensive solution and strong justification behind their approach.

The biggest challenge we encountered is that this particular situation does not satisfy the requirements for the application of traditional Bayesian statistics. Since the authenticator has no way to know how many total certificate chains may exist in the global TRG, traditional Bayesian statistics cannot be applied. Therefore, we turn to a broader area of study in human reasoning and adopt results from the area called *Scientific Confirmation Theory* [Car50]. Scientific Confirmation Theory is an area of study to model the reasoning behavior of humans. It deals with the situation where the assumptions of the traditional Bayesian statistics are not satisfied as in our case. A famous example of the *Raven's Paradox* and other details of scientific confirmation theory can be found in the [Car50, Mah93].

Combining Opinions from Multiple Certificate Chains

Our metric follows Reiter's suggestion [RS98] and considers only a set of node-disjoint certificate chains. Given a certification graph, there may be many ways to select a set of node-disjoint chains that have the same end nodes. It has been shown that the problem of finding all node-disjoint paths in a mesh graph is NP-hard [AKW96]. Reiter and Stubblebine also report on the difficulty

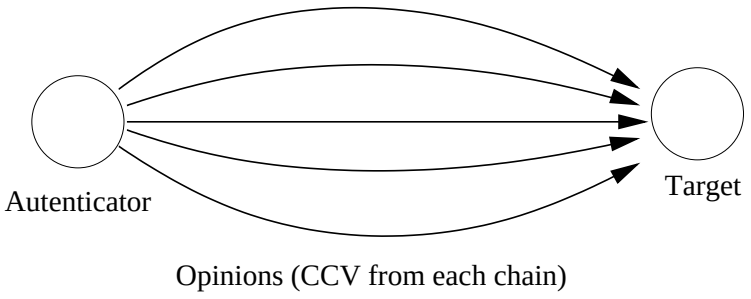


Figure 4.8: Combining Multiple Opinions

of enforcing the node-disjoint property on chain discovery. Therefore, we employ a simple greedy approach for chain discovery. The authenticating node, a, discovers *any* certificate chain from either itself or the trusted third party (if available) to the target node, t, in the local TRG. Finding a single certificate chain can be achieved by running a breadth-first search algorithm on the TRG. We use Dijkstra's algorithm to find the shortest chains available [Dij59]. After calculating the confidence value of the discovered chain, a removes the intermediate nodes of the chain from the local TRG and repeats this process until there are no more chains linking a and t. The confidence values from all discovered chains are collected and then combined into a single final QoA value at the end of the process. In the remainder of this section, the QoA calculation for each indirect authentication instance is presented in detail, as well as how these individual QoA results can be combined into a final QoA.

CCVs from each chain are considered an *Opinion* that the authenticator has about the target. Essentially, the authenticator has multiple opinions about a single target with the help of multiple disjoint sets of entities, as visualized in Figure 4.8.

To combine these potentially disagreeing opinions, our metric has two main steps. First, the CCVs from the chains must be combined. Second, the quality of the evidence given must be factored into the final decision of what QoA to assign to the target.

Given a number of CCVs, one per chain, the value of the mean (μ) is calculated according to Equation 4.4, where n is the number of chains.

$$\mu = \frac{\sum CCV}{n}. \tag{4.4}$$

The probability that an accurate QoA is established by the mean depends not only on the

sample size, but also in the variance between CCVs in the sample. Any CCVs that are far outside the mean skew the result. To avoid the potential of malicious nodes skewing the QoA by injecting a CCV at either end of the spectrum, a confidence interval is used to eliminate widely varying values. First, the sum of squares is calculated according to Equation 4.5. Then, using this value, the standard deviation (σ) is obtained according to Equation 4.6.

$$\sum d^2 = \sum CCV^2 - \frac{(\sum CCV)^2}{n}. \tag{4.5}$$

$$\sigma = \sqrt{\frac{\sum d^2}{n-1}}. \tag{4.6}$$

Finally, any CCV values outside two standard deviations away from the mean are removed from the sample and μ is recalculated using the new sample using Equation 4.4.

Next, the level of confidence in the value of μ based on the sample size is taken into account. Intuitively, a larger number of chains should yield a higher confidence in μ than a lower number of chains. Standard statistical methods for calculating the probability that μ is correct assuming that the number of possible samples is known ahead of time [Mah93]. In other words, if the possible number of samples is 100, we could calculate the probability that μ is correct given 40 chains. However, the number of possible chains cannot be not known due to the dynamic nature of ad hoc networks, barring us from using the standard statistical methods. To solve this problem, we turn to insights gained from the study of *Scientific Confirmation Theory*, an area of inductive logic [Car50, Mah93]. Essentially, we can think of the problem in the following way. Given a set of the n CCV values, we want to make a hypothesis about what the actual confidence value should be based on these evidences. Using a standard approach in scientific confirmation theory, we use Equation 4.7 to give the confidence factor (ω), where n is the number of reporting chains.

$$\omega = \frac{n}{n+1}. \tag{4.7}$$

Essentially, ω captures the level of confidence in the value of μ given the fact that n chains are being used for the calculation. The intuition behind Equation 4.7 is that the confidence in two chains should be significantly higher than the confidence in only one chain. However, the increase

in confidence from 99 chains to 100 chains should be quite small. This intuition is captured in the confidence factor ω by doubling ω from one to two chains, but only increasing ω by $\frac{1}{100}$ from 99 to 100 chains. Essentially, a low ω represents a low confidence in μ.

Once ω is determined from the sample size, it must be applied to μ to yield the final QoA. To control the impact of ω on μ, we introduce a weighting parameter γ as follows:

$$QoA = [(\mu \times \gamma) \times \omega] + [\mu \times (1 - \gamma)]. \tag{4.8}$$

Essentially, γ ($0.0 \leq \gamma \leq 1.0$), captures how much impact ω should have on μ by putting a cap on ω's range of impact on μ. The first term of this equation represents the portion of μ adjusted using γ, while the second term represents the unaffected portion of μ. For example, with $\gamma = 1.0$, ω is fully applied to the whole μ with maximum effect. If $\gamma = 0.5$, ω is only applied to 50% of μ and the effect is halved. When the sample size is very large, a small number of samples implies very incomplete knowledge. However, when the sample size is very small, a small number of samples still implies reasonably complete knowledge. Therefore, for a large sample size, (*e.g.*, 1,000), the value of γ should be high. However, for a small sample size, (*e.g.*, 10), the value of γ should be low. Once a final QoA has been calculated, the end user can decide whether or not to accept the authentication or reject it.

In the following three chapters, we present two novel situation-aware ad hoc key management frameworks: MOCA distributed PKI and Composite Key Management, followed by an extensive comparison study of existing ad hoc key management frameworks including our own.

Chapter 5

MOCA: A Secure Distributed PKI

Public Key Infrastructure (PKI) [KP] is a well-established approach for digital certificate management. PKI was originally designed around a centralized and trusted component called the *Certificate Authority (CA)*, which binds and unbinds entities to their public keys by issuing and revoking digital certificates, and also functions as the repository for active digital certificates. Well-known examples of certificate authorities for the Internet include Verisign [Ver], Thawte [Tha] and Entrust [Ent]. However, it is questionable if the traditional centralized CA-based PKI is directly applicable to ad hoc networks, where different characteristics invalidate many assumptions that traditional PKI relies on. First, nodes in an ad hoc network are more vulnerable compared to wired hosts. Second, network topology and connectivity can rapidly change in ad hoc networks due to the mobility of nodes and the use of a wireless medium, making it difficult to maintain the availability of a mobile CA. Therefore, an ad hoc PKI must be designed to operate under these unique conditions.

To address the challenges of ad hoc environments, the CA's functionality can be distributed to multiple nodes in such a manner that the CA stays secure even when some portion of the responsible nodes are compromised or become unavailable [ZH99]. Most approaches in this direction rely on a cryptographic technique called *threshold cryptography* [FD92]. However, there is an inherent tension between the two most important goals of a distributed CA: *strong security* and *high availability*. Careless focus on strong security can easily make the CA unavailable or too costly to be maintained. Similarly, a blind effort to increase the availability of a distributed CA can easily lead to a security breach of the CA. Therefore, an ad hoc key management framework must be designed as a comprehensive system addressing all of the challenges in a unified framework.

The contribution of our work in this chapter is the design and implementation of the MOCA (MObile Certificate Authority) ad hoc key management framework. MOCA uses threshold cryptography to distribute the CA functionality to multiple nodes. MOCA differs from other distributed CA approaches for ad hoc networks by limiting MOCA nodes to a small but more secure subset of the nodes in an ad hoc network. The distributed CA is then made available with novel communication support designed for the unique pattern that arises from the access to any threshold quorum system. These novel design characteristics enable MOCA to simultaneously provide strong security and high availability. MOCA is equipped with its situation-aware component that uses a novel combinatorial metric to measure the changing security level of distributed PKI so that the users of the service can adequately adapt to changing situations.

The rest of this chapter is organized as follows. We first discuss the threat model used in the design of the MOCA framework. Section 5.2 describes the MOCA framework in detail. In Section 5.3, we present our novel manycast routing protocol that enables MOCA to be efficient while maintaining strong security. We then analyze the security of the MOCA framework with our novel security metric in Section 5.4. In Section 5.5, the efficiency of communication support in the MOCA framework is illustrated with an extensive simulation study and we summarize the contributions in Section 5.6.

5.1 MOCA Threat Model

The threats that can exist in an ad hoc network must drive the design of a key management framework for ad hoc networks. Attacks can be classified into two categories: *active* and *passive*. Active attacks involve behaviors such as manipulating packets, attacking other mobile nodes, or jamming the wireless medium. Passive attacks only rely on overhearing the traffic without disrupting network operation. Passive attacks are harder to detect compared to active attacks with visible anomalies. We focus our attention on two active attacks on a distributed PKI: Routing layer attacks and Directed attacks on CA nodes.

- Routing Layer Attacks - Malicious nodes can disrupt routing behavior by advertising false routing information, injecting incorrect routing packets, or even luring all packets and drop-

ping them [AHNRR02, HPJ03a, HPJ03b, K. 02, YNK02]. Some routing layer attacks can be used to mount a simple denial-of-service attack if the attacker can either block or reroute all of the victim's packets. The MOCA framework uses a set of routing protocols based on the intelligent use of limited flooding that are immune to most routing layer attacks.

- Directed Attacks on CA nodes - When the attacker can discover either the identity or the location of CA nodes, the attacker can focus its resources on only attacking the CA nodes. The MOCA framework is designed to minimize the possibility of the CA nodes getting compromised. MOCA nodes are only selected from more secure and capable nodes and their identities are hidden so that an adversary cannot direct an attack on the MOCA nodes.

Radio frequency jamming is an active attack mounted at the physical layer of the network where an attacker transmits a high power signal over the spectrum, effectively *jamming* the band. This is a low-level denial-of-service attack and defense against it is out of scope of this study.

Passive attacks include eavesdropping and traffic analysis. The MOCA framework is not vulnerable to eavesdropping since all information contained in communication between a client and the MOCA framework is public. On the other hand, traffic analysis may provide the attackers with sensitive information about the configuration of the framework. However, it is unclear how feasible a traffic analysis attack can be in ad hoc environments since traffic analysis requires a large amount of the attackers' resources [JVZ00]. While there are known approaches to deter traffic analysis in wired networks [DMS04, RR98], it is questionable if any of these approaches can be directly applied to ad hoc environments due to their excessive communication and computation overhead. Therefore, the design of our MOCA framework focuses on the two specified active attacks. Based on this threat model, we next present a set of requirements for effective ad hoc key management frameworks and examine existing approaches based on these criteria.

5.2 MOCA

In this section, we present the MOCA (MObile Certificate Authority) ad hoc key management framework. MOCA uses threshold cryptography to divide and distribute the CA functionality to multiple nodes. MOCA nodes are picked carefully based on their characteristics such as physical

security, computational capability, and trustworthiness. Based on these criteria, a relatively small set of nodes is selected to function as the distributed CA. These MOCA nodes operate in the network without revealing their identity as the CA nodes. Careful MOCA node selection and anonymity help achieve a high level of security for the MOCA framework. Usually, the price for having a small set of CA nodes is reduced availability and higher communication overhead. MOCA addresses this problem by employing a suite of specially designed routing protocols for efficient communication support for the certification traffic.

5.2.1 Using Threshold Cryptography

MOCA employs *k-out-of-n* threshold cryptosystem to divide the CA's master private key to n distributed CA nodes [FD92, Sho00]. The CA's master private key is divided into n pieces and any k of these pieces can be used to reconstruct the master private key. There are three important factors to consider when employing threshold cryptography: (1) Who will be given a secret share? (2) Who will distribute the secret shares? And, (3) How to handle compromised secret share holders? Next, we discuss each of these questions in detail.

Choice of MOCA Nodes

Any number of nodes between one and the total number of nodes in the network can be selected as MOCA nodes. However, selecting a relatively small fraction of nodes in the network is desirable. Intuitively, the crypto threshold k cannot be set too high since it causes every certification request to generate excessive communication overhead. With the crypto threshold value fixed, increasing the number of MOCA nodes widens the gap between k and n, allowing attackers to more easily locate enough MOCA nodes to compromise and steal the master private key. Therefore, n must be kept to a relatively small value compared to the total number of nodes in the network.

There are many possible ways to select the MOCA nodes. While most research in ad hoc networking has implicitly treated all nodes to be identical, it is more likely that an ad hoc network contains several types of mobile nodes that are different from one another in power capacity, transmission range, computational capacity, and security. Therefore, any security service or framework should utilize this potential *heterogeneity*. For example, consider a battlefield scenario with a battle

group consisting of infantry soldiers, platoon commanders' jeeps, company commanders' command vehicles, artillery vehicles, transport vehicles, and tanks. All of these mobile nodes have different rank, power, computation capacity, transmission range, and level of physical security. In such a case, it would be wise to pick nodes with higher ranks, more power, more capabilities and stronger security to provide a security service. While it may not be necessary to exploit this potential heterogeneity to enhance basic ad hoc routing, certainly this heterogeneity can be used to help improve network security by endowing *more secure* nodes with sensitive information. Similar situations can be imagined in emergency rescue operations, disaster recovery, or any other scenario where ad hoc networks can play a critical role. In general, knowledge of such heterogeneity should be used to determine the nodes that share the responsibility of the CA. Once the number of MOCA nodes, n, is chosen, the MOCA framework selects n mobile nodes with (1) highest trustworthiness, (2) highest physical security, (3) most computational capacity, and (4) most power as MOCA nodes.

Once the MOCA nodes are selected, their identity must be kept secret. Maintaining the anonymity of MOCA nodes is crucial to achieve strong security. Intuitively, it is harder, if not impossible, for an adversary to locate anonymous MOCA nodes and compromise them. This forces the adversary to invest far more resources trying to compromise the key management framework and reduces the chance for successful attacks. We have previously proposed an anonymous communication protocol for ubiquitous environments called *Mist* [AMCK$^+$02]. Mist provides location privacy for communicating peers through support at the routing layer. Our approach for anonymous communication support in MOCA shares this design point and is designed as a routing layer solution. With routing layer support for anonymous communication and careful design of protocol message content, the MOCA framework provides certification service anonymously. A traffic analysis attack by a very powerful attacker may *guess* the identity of some MOCA nodes. However, due to the choice of MOCA nodes and the anonymous routing support, the attacker's guess cannot be verified and the anonymity of the MOCA framework only degrades a little.

Off-line Key Dealer

Once a set of nodes is selected to serve as MOCA nodes, the nodes are configured by an off-line key dealer that does not participate in the ad hoc network operation. The off-line key dealer generates

the CA's master key pair, divides the private key using threshold cryptography, distributes the key shares to selected MOCA nodes and then goes offline and stays out of the network. This off-line key dealer is a crucial component to the framework's overall security since it holds the full secret key of the CA. Therefore, it is critical to protect the off-line key dealer from attackers.

Alternative approaches that do not require such an off-line key dealer use online election of CA nodes. Such online election can be performed when more than a threshold number of CA nodes become disabled or there is a network configuration change. Kong et al._uses online election to make the distributed CA self-contained within an ad hoc network [KZL+01]. In their approach, a group of neighboring nodes can promote another node to become a CA node with a proper key share. However, this design decision opens the door for a serious security breach since it is relatively easy to compromise enough nodes to steer the online CA election to benefit attackers. Also, given the problem of Sybil attacks, it is impossible to prevent impersonation in an open distributed system without clearly distinguished centralized authentication support [Dou00]. Therefore, Kong et al.'s proposal is also open to Sybil attacks where an adversary node may acquire multiple identities to impersonate enough nodes to acquire key shares and reconstruct the CA's full private key, resulting in the total compromise of the framework. Defense against Sybil attacks in ad hoc networks is still an open research problem [NSSP04]. Therefore, MOCA does not allow online election of new CA nodes in the interest of maintaining strong security.

5.2.2 Message Format

Hiding the identity of the MOCA nodes is another important factor for strong security. It is also crucial that no message exchanged between a client and the MOCA nodes carry any identifying information about the MOCA nodes. There are two types of messages that can be exchanged between the MOCA framework and client nodes: certification requests from a client and certification replies to a client. All messages are designed so that the identities of MOCA nodes can stay hidden.

Certification Request (CREQ)

A certification request message is sent from a client node to a group of MOCA nodes. All MOCA nodes that receive the request message must reply accordingly. A certification request message

contains the following information.

1. Client ID - the ID of the client node
2. Type - A certificate request, a revocation request, or a certificate retrieval request.
3. Payload - A certification request contains the public key of the requesting node. A revocation request contains the public key of the requesting node. A certificate retrieval request contains the ID of a node whose certificate is being requested.
4. Signature - The whole certification request message is signed with the requesting node's private key. Note that this key may not yet be certified at this point.

Certification Reply (CREP)

When a MOCA node receives a certification request message from a client, it returns a certification reply message containing the following information.

1. Client ID - The ID of the requesting node
2. Payload - For a certification request, a partial signature over the digital certificate is sent back. For a revocation request, a revocation certificate is created and sent back. For a certificate retrieval request, a digital certificate is sent back if available at the MOCA node.

Note that a CREP message does not contain any information about the replying MOCA node to keep them anonymous.

5.3 Manycast Communication Support for the Certification Traffic

The design decision to limit the number of MOCA nodes in a network creates a challenge for availability. Since there are a limited number of MOCA nodes to choose from, a client node may have difficulty contacting enough MOCA nodes. Without any attention, this can also easily put excessive communication overhead on the network while causing network-wide congestion and

wasting the scarce resources of mobile nodes. The key idea of our solution lies in the observation of a novel communication pattern generated by the MOCA framework. When a client contacts a set of MOCA nodes for a certification service, it generates a communication pattern of *one-to-many-to-one*. The client needs to contact k MOCA nodes and must receive k independent replies. We named this communication pattern *manycast* and performed an extensive study of its characteristics [CYRK03].

Essentially, manycast is a group communication paradigm in which one client communicates simultaneously with some threshold number of servers from the members of a group. Manycast provides a unique communication challenge. As in anycast, the ideal set of receivers for a particular transmission varies according to its source. In fact, both anycast and multicast are special cases of manycast communication. To support service-oriented communication, manycast should enable efficient short transactional request/response communication between clients and servers, in addition to a one-way dissemination of data as in IP multicast. Due to the dynamic nature of ad hoc networks, the efficient support of this bidirectional one-to-many-to-one communication requires implementation in the network layer.

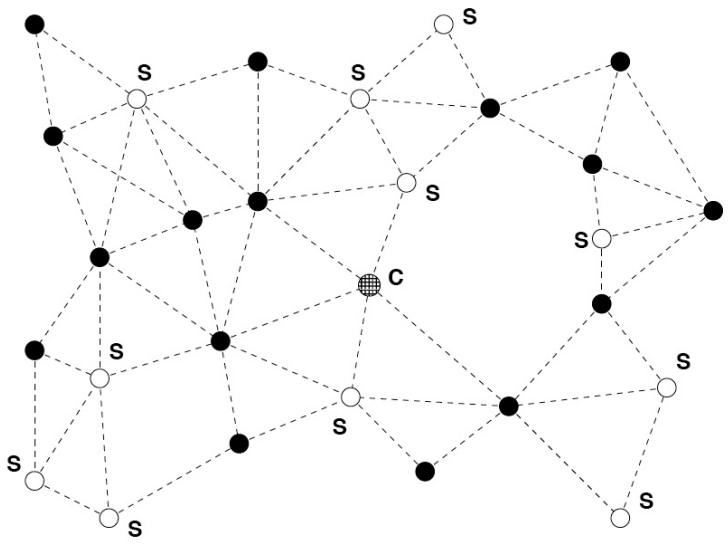
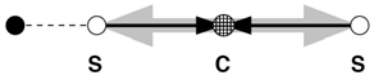


Figure 5.1: Example Ad Hoc Network

To provide a clear description of the manycast approaches, we define some necessary terminology and a simple graphical representation of communication patterns. Just as with unicast routing protocols, there are two phases to the routing process. In the discovery phase, the source has

no knowledge about the network and the targets of a transmission. In the delivery phase, the source has previously discovered something about the topology of the network and tries to use that knowledge to perform more efficient delivery than what is possible in the discovery phase. When the network knowledge is no longer useful, e.g., due to a link break, the routing process moves from delivery back into the discovery phase. We illustrate the operation of each approach using the ad hoc network of 28 nodes in Figure 5.1. This network supports a manycast application with one client node, the hatched node labeled C, and 11 servers, the white nodes labeled S. The goal for this application is to reach 3 servers with each manycast transaction. The dashed lines between nodes indicate connectivity. We indicate request and relay transmissions (those that deliver the request to servers) as thick, lightly shaded directed edges. Transmissions that carry the response message back to the client are depicted as narrower, darker directed edges. A depiction of these edge styles illustrates the representation:



When a transmission is sent in both directions across a link during the course of a scenario, we remove the arrowheads and draw a single undirected edge between the two nodes. We indicate broadcast transmissions with multiple outgoing edges from the same node. The description in the text clearly differentiates whether multiple outgoing edges indicate a single broadcast transmission or several unicast transmissions.

An ideal manycast delivery has no discovery phase and requires the smallest possible number of transmissions to perform a transaction. For the example network, an ideal manycast delivery is presented in Figure 5.2. This delivery reaches exactly 3 servers. All transmissions, with the exception of the original request, are unicast.

When designing a protocol to support manycast communication, there are two questions to answer: (1) How to choose and contact k nodes from the set of n without knowing their individual identities? And (2) What is the most efficient way to contact and receive individual replies from them? Among the suite of manycast routing protocols proposed in [CYRK03], only two are appropriate for the MOCA framework: *flooding* and *scoped-flooding*, since only these two approaches can maintain the servers' anonymity.

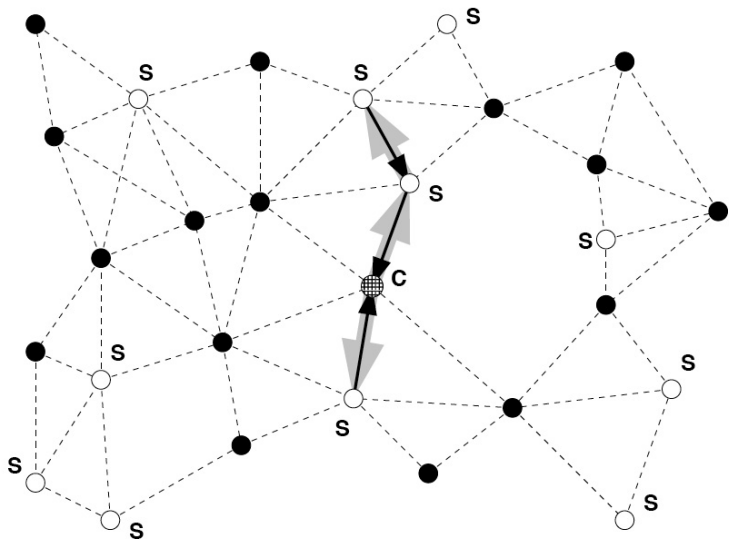


Figure 5.2: Ideal Manycast

A client without any knowledge about the network or MOCA nodes must flood a certification request (CREQ) the first time. Every MOCA node that receives the CREQ replies with a certification reply (CREP) generated with their secret share. A CREP message is unicast back to the client using the reverse path created by the CREQ message. When a client receives CREP messages, the client records the hopcount of each reply. These hopcounts show the distance between the client and the MOCA node that sent the reply messages. When the client needs to send the next CREQ message, the information in this hopcount cache is first examined to find out whether it is possible to reduce the scope of flooding controlled by the TTL (Time-To-Live) field in the CREQ message while reaching enough MOCA nodes. With scoped flooding, the MOCA framework achieves highly efficient communication support for certification traffic while keeping the MOCA nodes secure and anonymous. A detailed performance study of manycast routing in the MOCA framework is presented in Section 5.5.

5.4 Security Analysis

In this section, we examine the security of the MOCA framework. Threshold cryptography used to distribute the CA functionality has built-in support for security and fault tolerance. More specifically, an adversary must compromise at least k MOCA nodes to compromise the CA. As

long as there are k non-faulty MOCA nodes in operation, the framework can provide service. However, deployment of a distributed CA with threshold cryptography requires careful attention to finer details. We first examine the basic parameters and their relationship to the security of the framework and then discuss some additional precautions required for secure deployment.

5.4.1 Threshold Cryptography Parameters

The configuration of the MOCA framework is determined by the total number of nodes in the network, M, the crypto threshold for secret reconstruction, k, and the number of MOCA nodes, n. While M cannot be chosen *a priori*, the crypto threshold, k, can be selected and it is important to understand the effects of the selected value of k. k can be chosen between 1 (a single MOCA node must be contacted for a certification service) and n (all MOCA nodes must be contacted for a certification service). Setting k to a higher value has the effect of making the system more secure since k is the number of MOCA nodes an adversary needs to compromise to penetrate the system. But at the same time, a higher k value makes clients contact more MOCA nodes for certification service, which may result in higher communication overhead. Therefore, the choice of k must strike a balance between the two conflicting goals by being small enough to not overwhelm the network but large enough to withstand attacks.

The number of MOCA nodes, n, is determined by the characteristics of the nodes in the network and is determined before the MOCA framework is deployed. n can be changed to a new value but it requires costly intervention of the off-line key dealer. In a threshold system, n defines the limits of the system as an upper bound for k since $1 \leq k \leq n$. Given a fixed value of k, a larger n increases the availability of the whole framework since a client can choose from a larger set of MOCA nodes. On the other hand, *(n - k)* is the maximum number of faults the framework can survive, so a larger n also means higher fault tolerance.

5.4.2 Measuring the Security Level for Distributed CAs

In this section, we briefly present the metric for CA's security level from Section 4.2.1 and analyze general distributed PKIs using the metric Assuming the distributed CA nodes are anonymous and an adversary cannot discover their identity, the best approach for the adversary is to compromise

as many nodes as possible in a given amount of time, hoping that enough CA nodes are included among the compromised nodes. The following simple combinatoric equation captures this situation.

$$\text{Security Level} = 1.0 - \frac{\sum_{i=k}^{c} \binom{n}{i}\binom{M-n}{c-i}}{\binom{M}{c}},$$

This formula measures the probability that an attacker fails to compromise the distributed CA given that the attacker can compromise at most c nodes in a window of time. If an attacker is capable of pinpointing attacks only on the CA nodes, the attack always succeeds as long as $c \geq k$. Therefore, it is crucial to keep the CA nodes anonymous, limiting attackers to random attacks.

Currently, there are only two concrete designs proposed for distributed CAs [KZL+01, YK03]. In Kong's approach, every node serves as a CA node. Therefore, it is impossible to hide the identities of CA nodes and application of this metric always yields a zero security level as long as $c \geq k$. In contrast, MOCA hides the CA nodes' identities as well as limits their number. Therefore, the best chance an adversary has to compromise the system is by randomly compromising as many nodes as possible. For example, the security level of a 30-node MOCA framework with $k = 5$ and $c = 10$ in a 150-node network is calculated to 0.97, which shows that it is not very likely that this configuration of the MOCA framework will be compromised.

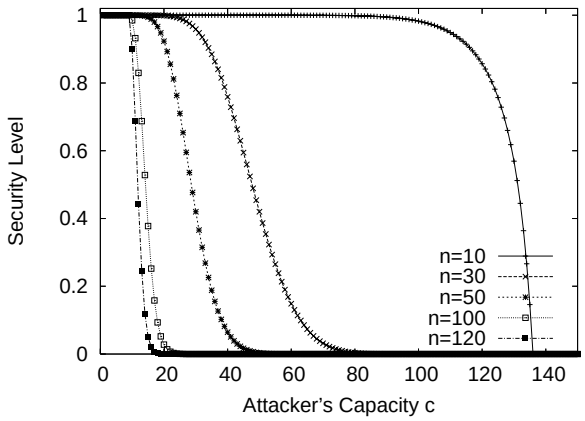


Figure 5.3: Security Level with Varying Number of MOCA Nodes n (it k=10)

This metric also reveals another important configuration pitfall that can be easily overlooked: the meaning of the gap between k and n. Based on the discussions so far, a large enough k and much larger n appear to be the right choice for configuration parameters. However, if the gap between

k and n is too big, the security of the framework degrades. If $c < k$, the framework is secure by design since no adversary can compromise enough MOCA nodes. However, if $c \geq k$, it is possible for an adversary to compromise the framework. Therefore, it helps to limit the MOCA nodes to be a small set of more secure and capable nodes, which makes it harder for the adversary to locate and compromise enough MOCA nodes. Figure 5.3 illustrates one example. Out of 150 total nodes in the network, 10, 30, 50, 100, and 120 nodes are selected as MOCA nodes with a fixed crypto threshold $k = 10$. The five curves in the graph display cases of $n = 120$, 100, 50, 30, and 10 from left to right. As the attacker's capacity c increases, all curves monotonically decrease from 1.0 to 0.0. However, the rate of decrease is much higher for curves with a larger n. This illustrates the effect of the gap between k and n, which shows that a too large n can weaken the configuration of the overall framework.

This metric introduces a new burden on end users who must apply this metric to measure the security of distributed CAs. Each end user must be able to determine an adequate c value for the attacker(s)'s capacity under changing network condition. End users can collect relevant information from available support facilities such as intrusion detection systems or any other monitoring service provided by network operators and also use his own perception on the network conditions to determine the adequate c value. Therefore, end users can have different c values based on their own perceptions. However, determining the accurate value of c under changing network condition is still an open research problem that we plan to investigate in the future.

5.4.3 Selection of MOCA nodes

As discussed in the previous section, the number of MOCA nodes, n, in a network should be limited to a reasonable number. It may seem counterintuitive to limit the number of MOCA nodes, which may reduce the availability and the fault tolerance achieved by the distributed nature of MOCA. For example, in a 300 node network, an operator may have a choice of selecting 200 random nodes or 30 nodes with higher physical security to support CA functionality. Blindly comparing the number of MOCA nodes in the system, the first choice seems better because it has more MOCA nodes in the network, improving fault tolerance and availability. However, by guaranteeing a higher level of security of the 30 MOCA nodes in the second case, compromising them becomes much harder

than compromising the randomly selected MOCA nodes in the first case, hence making the second case more secure against adversaries. It is possible that an ad hoc network does not have enough heterogeneity among the nodes, which may make it difficult, if not impossible, to choose MOCA nodes based on heterogeneity. In such cases, we can fall back to random selection of MOCA nodes. However, the level of security will decrease since there is no guarantee on the security of each MOCA node.

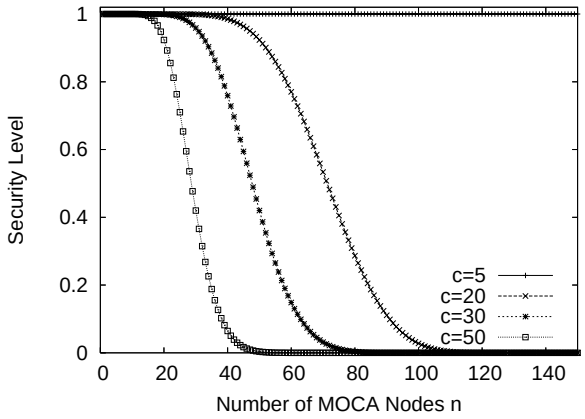


Figure 5.4: Security Level with Varying the Attacker Capacity c (k=10)

The next question is how to pick the best subset of nodes to serve as the MOCA nodes. As discussed in Section 5.2.1, MOCA nodes are selected based on their characteristics and limited to more secure, more capable, and more trustworthy nodes. This design decision makes it more difficult for an adversary to compromise the MOCA framework, in effect decreasing the adversary's attack capacity c. Figure 5.4 illustrates an example. The three curves display the security levels for the configurations with $c = 50$, 30, and 20. The horizontal line at the security level 1.0 is for the case of $c = 5$, where the framework is completely secure since $c < k$. The curves move to the right as the adversary's attack capacity is decreased, making the system more secure. A careful selection of MOCA nodes can indeed help maintain a higher security level.

5.4.4 Degradation of Threshold Cryptography

MOCA nodes are selected off-line and are not resurrected or re-elected during operation. It is possible to elect nodes to become MOCA nodes on-line during operation of the network but it may lead to a security breach. If the system supports online election of CA nodes and an adversary

successfully acquires k identities, that adversary can be elected to be the CA node k times, effectively recovering the CA's secret key. Therefore, MOCA chooses not to support any replenishing of CA nodes but degrades gradually until less than k CA nodes are alive. At this point, the off-line key dealer intervenes to reconfigure the framework, potentially suspending the normal network operation for a short period.

Online secret share update has been proposed by Zhou to allow as many of the distributed CA nodes as possible to be updated and remain secure and available for service for longer time [ZSvR]. However, the scheme is known to generate excessive communication overhead and heavy computational burden and also has never been applied to an ad hoc environment. We believe it is not suitable for an ad hoc environment where both communication and computation uses scarce battery power and other resources.

5.5 Communication Performance Evaluation

The focus of our performance evaluation is to measure the effectiveness and efficiency of MOCA communication support. We show that the MOCA framework can maintain a secure distributed CA without incurring prohibitive communication overhead to cripple normal network operations by employing scoped-flooding. The effectiveness of the framework is measured by the success ratio of certification requests. Given a crypto threshold k, more than k replies from MOCA nodes makes a certification request successful. The success ratio must be kept at a high level under all circumstances to provide useful service. However, a high success ratio should not come at the price of excessive overhead that can affect normal network operation. Overhead is measured by the number of messages transmitted per certification request. The simulation results show that scoped-flooding achieves a very high success ratio comparable to pure flooding with an acceptable packet overhead.

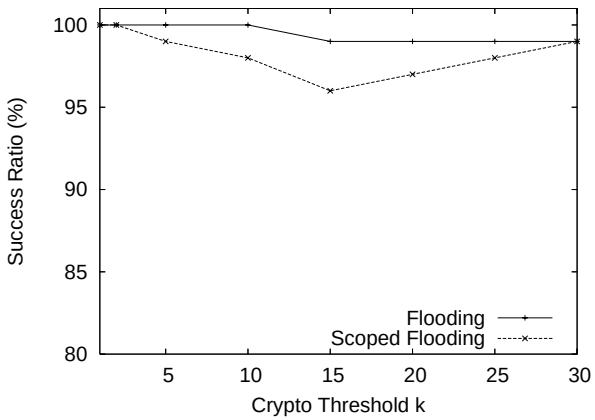
5.5.1 Simulation Set-Up

We implement our certification protocols in the ns-2 network simulator [ns2]. 150 mobile nodes are set up within either 1km by 1km area or 2km by 2km area. Out of 150 nodes, 30 nodes are randomly selected as MOCA nodes. 30 MOCA nodes represent 20% of the total nodes, which we

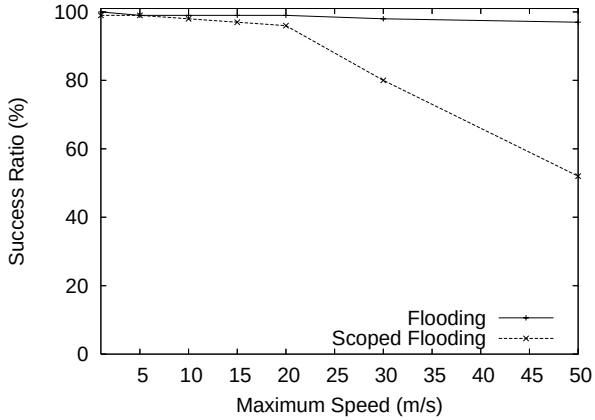
believe provide a reasonable number of MOCA nodes to support the given ad hoc network. Each simulation is run for 600 seconds. Detailed simulation parameters are listed in Table 5.5.1. The certification request pattern includes 100 non-MOCA nodes, each making 10 certification requests randomly distributed through the simulation timeline, for a total of 1000 certification requests. Each requesting node makes one request per minute on average during the course of the simulation. This is roughly 100 requests per minute and we believe that this is a reasonable number if not too stressful to the framework. Assuming each certification request precedes initiation of a new secure communication, starting one secure communication session per node per minute should be more than adequate for ordinary mobile nodes. Node movement follows the random waypoint mobility model implemented by Yoon et al. [YLN03]. Data points in the graphs are averaged over five different mobility scenarios with identical simulation parameters. We measure the performance of scoped-flooding with pure flooding as the baseline since they are the only anonymous manycast routing protocols available. Pure flooding always floods the network with certification requests, potentially incurring high overhead. In comparison, scoped-flooding uses a limited flooding of certification requests with a reduced TTL value when there is enough cached information. Scoped-flooding only falls back to pure flooding when there is not enough information in the hopcount cache. For all simulations, we control two parameters that affect the performance of the MOCA framework: the crypto threshold k and the mobility of nodes as measured in maximum speed.

- Crypto Threshold k - k is the minimum number of CREPs required for a client to reconstruct the MOCA's full signature and render the certification request successful. If k is set to a small number, a client only needs to collect a small number of k partial signatures to continue. Therefore, with a small k, the success ratio increases and the packet overhead decreases. A large k value makes attacks more difficult, but the burden on clients and the packet overhead increase since a client needs to contact a large number of MOCA nodes for a certification request.

- Mobility (Maximum Speed) - As nodes move faster, it becomes harder to maintain connectivity to enough MOCA nodes. When scoped-flooding is used, the client relies on its previous knowledge about the number of nearby MOCA nodes. Under high mobility, this knowledge remains valid only for a short period and scoped flooding fails more frequently, resulting in



(a) Varying k (Max. Speed 20 m/s)



(b) Varying Mobility ($k = 15$)

Figure 5.5: Success Ratio in the 1000m x 1000m Scenario

decreased success ratio.

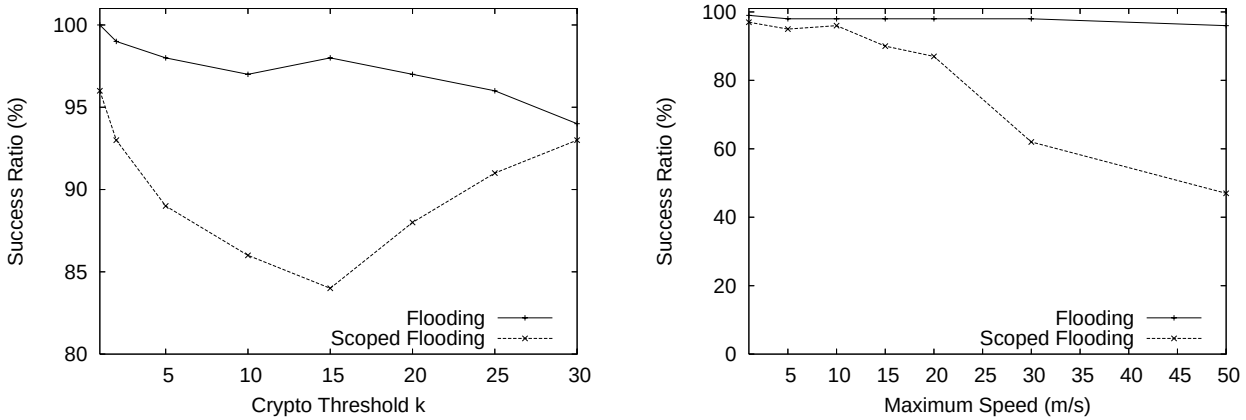
Our simulation results show consistent patterns throughout different pause times, speed patterns and number of MOCA nodes. Therefore in this section, we only present the results for varying k with a fixed maximum speed of 20 m/s and varying maximum speed with a fixed $k = 15$.

Total Number of Mobile Nodes	150
Number of MOCA nodes	30
Area of Network	1000m x 1000m, 2000m x 2000m
Total Simulation Time	600 sec.
Number of Certification Requests	10 requests each from 100 non-MOCA nodes
Node Pause Time	0, 10 sec.
Maximum Node Speed	0, 1, 5, 10, 15, 20, 30, 50 m/s
Crypto Threshold k	1, 2, 5, 10, 15, 20, 25, 30

Table 5.1: Simulation Parameters

5.5.2 Success Ratio

Success ratio for pure flooding stays higher than 98% under all crypto threshold values in the 1km by 1km scenario, showing the effectiveness of flooding as a manycast communication protocol (Figure 5.5 (a)). Scoped-flooding also maintains a high success ratio between 96% and 99%. The success ratio for scoped flooding is at its lowest with $k = 15$ when many scoped-flooding attempts fail because of the stale information in the hopcount cache. Success ratio again increases as k grows



(a) Varying k (Max. Speed 20 m/s) (b) Varying Mobility ($k = 15$)

Figure 5.6: Success Ratio in the 2000m x 2000m Scenario

larger than 15 because there is not enough information cached at the client and the client falls back to pure flooding. A similar pattern is amplified in Figure 5.6 (a) since the larger area and the lower node density affect the success ratio adversely.

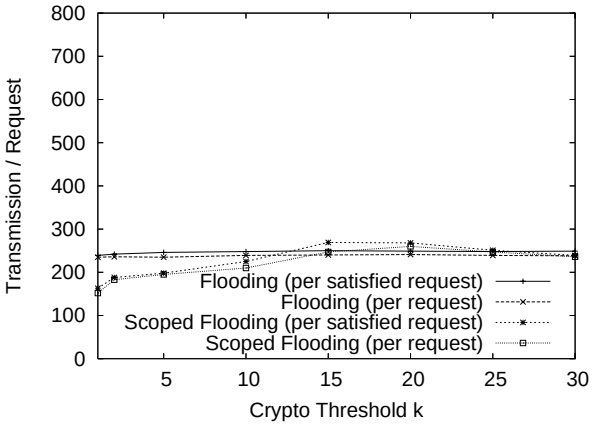
Both flooding and scoped-flooding are not affected by the mobility until the maximum node speed reaches 20 m/s. With mobile nodes traveling faster than 20 m/s, the success ratio of scoped-flooding degrades down to 52% under an extreme maximum speed of 50 m/s (Figure 5.5 (b)). This shows the effect of having stale information in the client's cache due to high mobility, which results in more failed scoped-flooding attempts. Figure 5.6 (b) shows a similar pattern in the 2km by 2km scenario.

The effectiveness of the MOCA framework is demonstrated with these results. MOCA is capable of providing almost perfect availability under reasonable network conditions and the performance gradually degrades as the network condition becomes pathological.

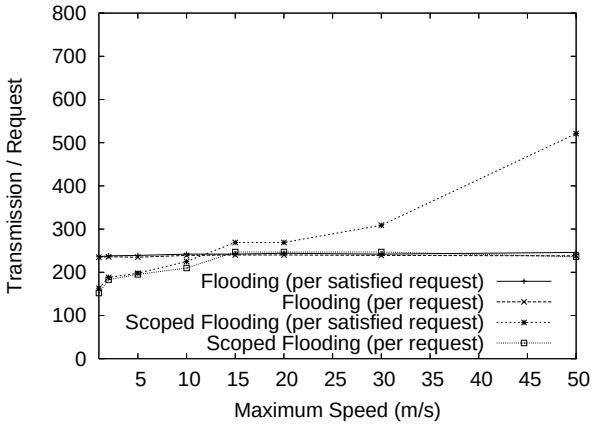
5.5.3 Packet Overhead

To measure the packet overhead for the MOCA framework, we measure the number of total packets transmitted per request. When a packet is broadcast, it is counted once per hop. Unicast packets are counted at each hop. While the number of packets per request simply measures the amount of packet overhead, the number of packets per *satisfied* request shows the effect of the success ratio.

In Figure 5.7 (a), both flooding and scoped flooding show little difference under all k values

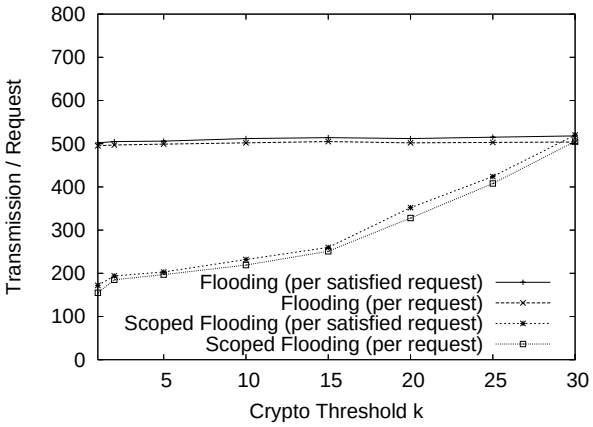


(a) Varying k (Max. Speed 20 m/s)

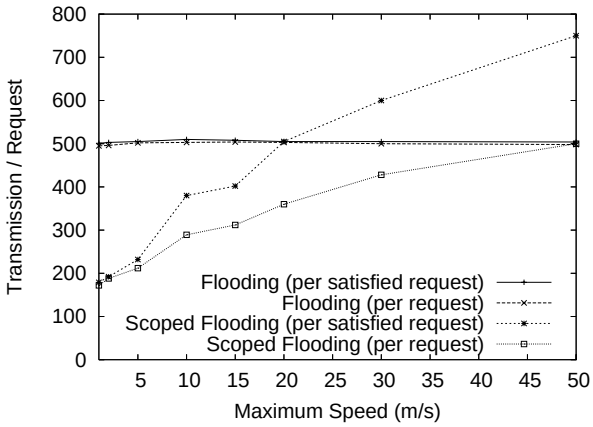


(b) Varying Mobility ($k = 15$)

Figure 5.7: Packet Overhead in the 1000m x 1000m Scenario



(a) Varying k (Max. Speed 20 m/s)



(b) Varying Mobility ($k = 15$)

Figure 5.8: Packet Overhead in the 2000m x 2000m Scenario

in the 1km by 1km scenario. For both cases, the number of packets per *satisfied* request is a little higher than the number of packets per *all* requests since only a small portion of certification requests fail. However, Figure 5.8 (a) shows the power of scoped flooding in a larger area. The packet overhead of scoped-flooding is less than half of pure flooding in the lower k range. This shows the effect of localizing manycast transactions by scoped-flooding, which improves the scalability of the overall framework. As scoped flooding fails more often with k larger than 15, the overhead catches up to pure flooding.

While flooding shows a similar result under varying mobility in Figure 5.7 (b), the number of packets per satisfied request for scoped-flooding increases to higher than 500 packets per satisfied request. This shows the limitation of scoped-flooding to cope with very high mobility. Figure 5.8 (b) displays a similar pattern in a larger area. Again, scoped-flooding performs better in the lower mobility range but the overhead per satisfied request grows very quickly as mobility increases.

These results show that MOCA's scoped-flooding can effectively suppress the packet overhead by limiting the certification traffic to local regions. In small scale scenarios with 1km x 1km area, scoped-flooding incurs a little lower overhead than pure flooding. However, as the network size grows, scoped-flooding successfully suppress the overhead explosion from pure flooding. Scoped-flooding is a highly efficient and scalable approach to provide manycast communication support.

5.6 Summary of Contributions

In this chapter, we present MOCA, a practical key management framework for ad hoc wireless networks. We clarify the necessity and the challenge of providing a PKI framework for ad hoc networks and identify the requirements for such a framework. Based on our observation of the potential heterogeneity among mobile nodes, we provide an intelligent way to pick a set of CA nodes. These selected secure nodes are called MOCA nodes and share the responsibility of collectively providing the CA functionality for an ad hoc network without revealing their identity. To achieve both strong security and high availability of the MOCA framework, we provide insight into the secure configuration of threshold cryptography and the observation of a novel communication pattern named *manycast*. To minimize the usage of scarce resources in mobile nodes, we develop a set of efficient and effective manycast communication protocols for mobile nodes to correspond with

the MOCA framework. Our security analysis shows that the MOCA framework can be configured to defend against capable attackers and our simulation results show the effectiveness of manycast communication support. As shown in Figure 5.9, the MOCA framework can be placed very high on the QoA axis and covers a range between medium to high availability for success ratio axis compared to other existing approaches. In the next chapter, we present a novel approach for ad hoc key management that improves MOCA's availability challenge.

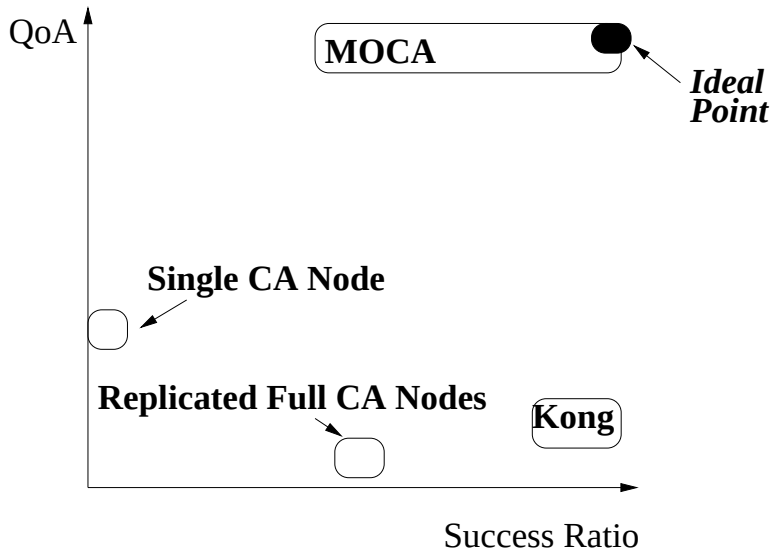


Figure 5.9: Comparison between MOCA and Existing Ad Hoc PKIs

While the MOCA framework's performance in QoA is very close to the ideal point, there is much to be improved on the success ratio. The wide range of success ratio achieved by the MOCA framework means that there can be situations when the MOCA framework is unavailable for service. In the next chapter, we present a novel Composite Key Management framework that addresses this availability challenge without compromising the high QoA of the MOCA framework.

Chapter 6

Composite Key Management

In the previous chapter, we presented the MOCA distributed PKI for ad hoc environments. MOCA provides high quality authentication service based on an intelligent use of threshold cryptography and anonymous *Manycast* routing support. Only shortcomings of the MOCA framework is that it may not be always available for service under rapidly changing network conditions, which affects the overall quality of authentication service the MOCA framework can provide.

In comparison, certificate chaining uses a very different approach to key management. Due to its decentralized nature, certificate chaining can provide excellent availability. But the quality of the provided authentication is not very high due to its dependence on the voluntary actions of ordinary mobile nodes. Certificate chaining fits naturally with ad hoc networks where there is no physical infrastructure, relying on each mobile node to issue certificates to other nodes at their own discretion. Certificate chaining requires a warm-up period to populate the certification graph, which completely depends on the individual node's behavior and mobility. Additionally, there are no guarantees that the resulting certification graph will be dense enough to be useful. Finally, the validity of a certificate chain depends on the trustworthiness of all the mobile nodes in the chain, which may not be easy to ensure in open networks. This dependence on potentially unknown nodes and the lack of any trust anchor in the system make certificate chaining unsuitable for situations requiring strong security guarantees.

While both MOCA PKI and certificate chaining have different advantages and limitations, neither approach is effective in all scenarios. To address their limitations in context, we define two underlying principles for providing secure key management in ad hoc networks. First, the burden of key management should be distributed to all nodes. Essentially, the more nodes participating

in key management, the more available the framework. However, it is important to distribute key management functionality in a way that maintains a high level of security. Second, it is highly beneficial to provide a trusted third party as a trust anchor for the network. Without a trust anchor, the quality of authentication cannot exceed a certain level. The presence of a trusted third party can significantly increase the quality of authentication.

To take advantages of the benefits of both techniques and satisfy these principles, we propose *Composite Key Management*, which simultaneously deploys multiple key management mechanisms, including distributed CAs and certificate chaining. By combining the characteristics of both of these mechanisms, Composite Key Management can provide high quality authentications with a high level of security and almost ubiquitous availability. A Composite Key Management framework can also adapt to dynamic changes in the availability of key management services. For example, Composite Key Management can provide excellent service in a network that supports both a distributed CA and certificate chaining. However, if one of the services is not available to a node, the node can still use the remaining services to receive the best possible authentication service. Essentially, users have a full spectrum of choices in how to participate in and use the service.

To complete our framework, we present an authentication metric to determine the trust level of the authentication and evaluate it as compared to pure distributed CA or pure certificate chaining frameworks. These evaluations demonstrate that Composite Key Management can be used to augment both distributed CAs and certificate chaining, improving both the success ratio for authentication and the level of confidence in the authentication.

The rest of the chapter is organized as follows. The detailed design of the Composite Key Management frameworks is described in Section 6.2. Section 6.3 presents the evaluations of Composite Key Management with comparisons to existing approaches. Finally, we summarize the contributions in Section 6.4.

6.1 Composite Key Management

It is apparent that an effective key management framework for ad hoc networks must include a secure TTP but still encourage participation from as many nodes as possible. To address both of these principles, we propose a novel paradigm for ad hoc key management called *Composite*

Key Management, which uses a distributed CA and certificate chaining simultaneously in a single ad hoc network. The distributed CA in Composite Key Management follows the design decisions from the MOCA distributed PKI and uses only a small subset of more trustworthy and secure nodes for the distributed CA. With this design, Composite Key Management can provide a TTP with strong security, satisfying the use of a TTP principle and the security component of the node participation principle. At the same time, the rest of the nodes participate in certificate chaining along with the distributed CA nodes to satisfy the availability component of the node participation principle, improving the availability and the coverage of the distributed CA to a level of ubiquitous presence. This combination of mechanisms can also improve the quality of authentication over pure certificate chaining since a certificate chain-based authentication can now rely on the TTP as a trust anchor, making the authentication inherently more trustworthy. It is important to note that while Composite Key Management improves availability over a pure distributed CA approach, the quality of the authentications that include certificate chains are lower than those only using a distributed CA. However, this reduced quality of authentication only applies to the requests that would have failed completely without Composite Key Management.

However, it is not simple to combine two heterogeneous approaches into a unified framework. Essentially, the meaning of an authentication result becomes more complex since end users must understand two different types of mechanisms and reason about interactions between them. To solve this problem, we utilize the concise set of authentication metrics from Sections 4.2.1 and 4.2.2 that encompass both distributed CAs and certificate chaining as well as the interactions between them. With this metric, an end user can easily calculate a trust value for a given authentication request to render decisions about whether or not to authenticate another node. Details of Composite Key Management is presented in Section 6.2.

The situation diagram for Composite Key Management is essentially the combination of situation diagrams of a distributed PKI and certificate chaining since a composite framework combines the two mechanisms.First two network properties, Secure Servers and Connectivity to Secure Servers are from the PKI component and the other two network properties, Trustworthy Mobile Nodes and Dense Certification Graph are from the certificate chaining component. The environmental factors that affect each network property remains the same. It is shown in Figure 6.1.

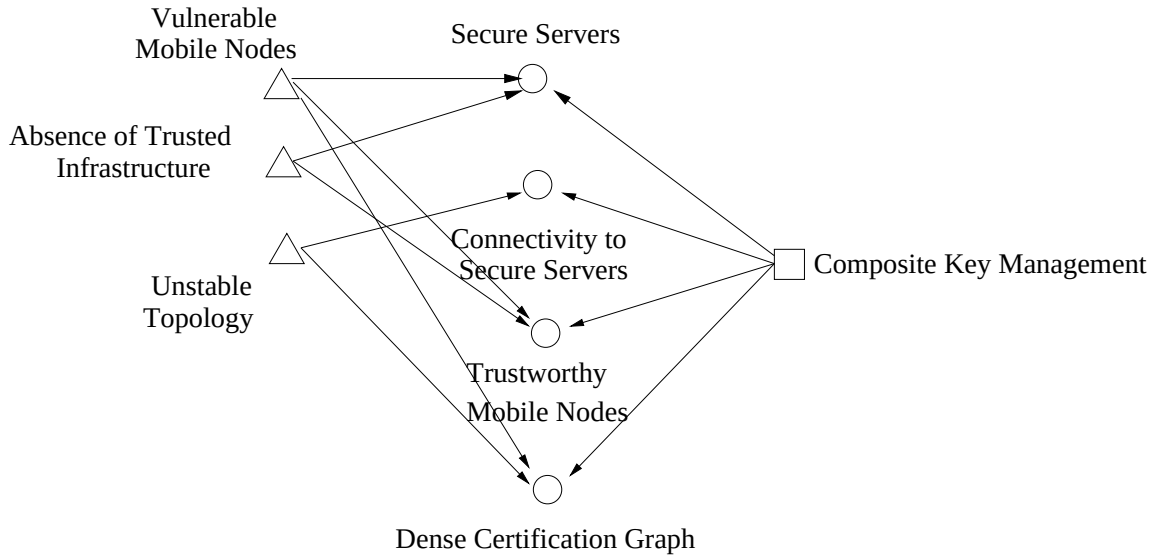


Figure 6.1: Situation for Composite Key Management

6.2 Design of Composite Key Management

The composition of a distributed CA and certificate chaining requires deploying the two frameworks simultaneously and equipping the end users with the proper tools to use the composite framework. The metrics of authentication presented in Section 4.2.2 allow end users to understand potentially complicated authentication information provided by a Composite Key Management framework. We can view the metric as the "glue" that binds all the components together. Since a distributed CA and certificate chaining are both self-contained approaches, they can be deployed in any manner possible: simultaneous deployment of both, adding a distributed CA to an existing certificate chaining system, or adding certificate chaining to an existing distributed CA framework. To better understand the interactions among the nodes in a composite framework, we first describe the three different types of nodes and clearly define their roles. We then list some specific examples of Composite Key Management.

6.2.1 Node Types in a Composite Key Management

In Composite Key Management, there are three types of nodes : CA nodes, nodes participating in certificate chaining, and client nodes that use the key management service. A single node can belong to more than one group.

- *CA Node*: A CA node carries a share of the distributed CA's private key and serves as one of the multiple nodes that comprise the distributed CA. A CA node is equipped with the capability to generate partial signatures using its key share, participate in certificate revocation and maintain a list of certificates issued by the distributed CA. For a detailed example of this type of node, we refer readers to previous works on distributed PKI in ad hoc networks [KZL+01, YK03, ZH99].

- *Participant in Certificate Chaining*: A node participating in certificate chaining must be able to authenticate its neighbors, create and issue certificates for neighbors, and maintain the set of certificates it has issued. For a detailed example of this type of node, we refer readers to Hubaux et al. [CBH03].

- *A Client*: Any client that makes authentication decisions must be able to understand certificates from both the distributed CA and from certificate chaining. Therefore, all client nodes must be equipped with the metric of authentication presented in the previous section. All authentication information is mapped to a local certification graph, which is used, along with the metric of authentication, by the client to calculate a confidence value for an authentication instance and decide on the authentication of the target node. This type of decision process allows individual nodes to apply their own criteria as to whether or not to authenticate on a per authentication basis.

6.2.2 Composition Examples

Since Composite Key Management currently utilizes two types of techniques, it is useful to separate the effects of each technique on the other and study them in isolation. Therefore, we present example compositions based on each technique. By gradually adding in the other technique, we can observe the effects separately. Since the composition examples use a distributed CA and certificate chaining, there are two base certification graphs that need to be composed. Figure 6.5 (a) represents the certification graph for the distributed CA component. All edges begin at the CA node and end at the end user nodes. Additionally, all edges are solid, indicating that these edges represent CA-issued certificates. Figure 6.6 represents the certification graph for the certificate chaining component. All edges are dashed arrows representing certificates are issued by peer nodes. These distinctions

between the two types of edges are only for illustrative purposes and edges are not distinguished in the actual application of the metric.

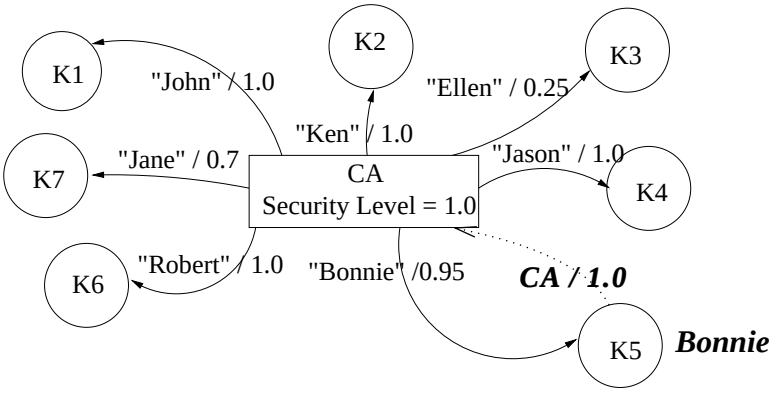


Figure 6.2: Certification Graph for Typical CA Approach

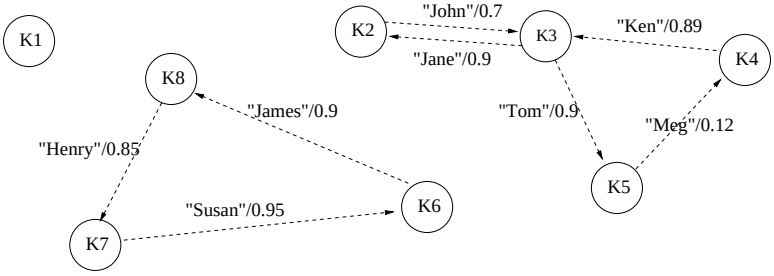


Figure 6.3: Certification Graph for Certificate Chaining

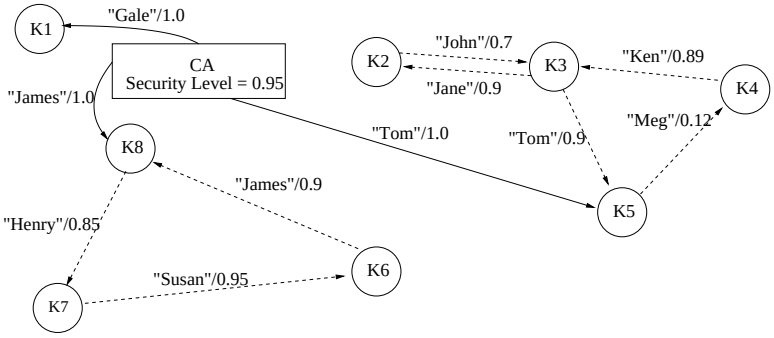
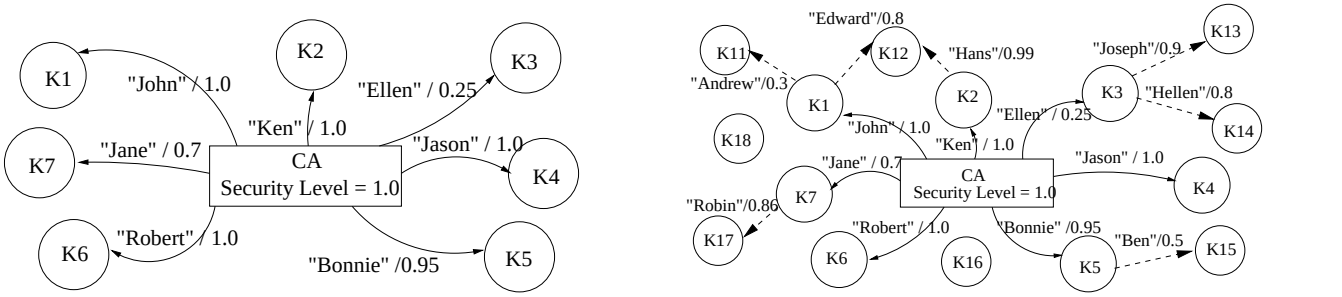


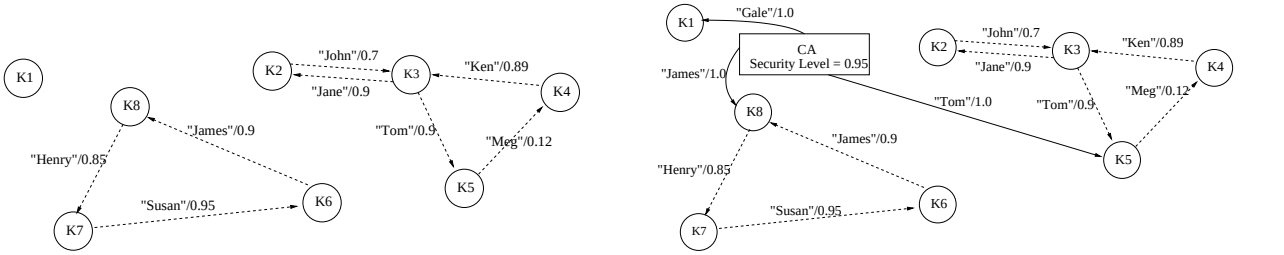
Figure 6.4: Certification Graph for Typical Composite Approach

The first composition uses certificate chaining to enhance the coverage of a distributed CA. The configuration of the certificate chaining component determines the limit on chain lengths. With 1-hop chaining, only nodes that have been certified by the distributed CA are allowed to issue certificates to other nodes. In this configuration, if a node wishes to acquire a certificate but



(a) Plain Distributed CA (b) Distributed CA composed with 1-hop Certificate Chaining

Figure 6.5: Distributed CA composed with 1-hop Certificate Chaining



(a) Plain Certificate Chaining (b) Certificate Chaining composed with a distributed CA

Figure 6.6: Certificate Chaining composed with CA-certified Nodes

cannot reach the distributed CA, the node can search its neighborhood to find any node that has been certified by the distributed CA. The original distributed CA certification graph in Figure 6.5 (a) is augmented with several 1-hop chains to form the graph in Figure 6.5 (b). Nodes with incoming dashed edges, like K11 and K12, are certified by CA-certified nodes, but not by the CA. In the original distributed CA certification graph, the average confidence value for all nodes is 0.843. In the composed graph, while the extended coverage of the composite framework covers six more nodes (K11, K12, K13, K14, K15, K18), the average confidence value decreases to 0.657 (with $p = 0.1$ for attenuation) due to the lower confidence values of the newly added certificate chains.

The second composition begins with a pure certificate chaining component. A TTP is introduced by allowing CA-certified nodes to participate in certificate chaining. By design, a node certified by a CA is more trusted and can be used to create new chains with higher levels of assurance. The certification graph of the pure certificate chaining component in Figure 6.6 (a) can be augmented with certifications from a distributed CA as shown in Figure 6.6 (b). In the original certification graph in Figure 6.6 (a), there are three SCCs and nodes can authenticate each other only within an

SCC. For example, K7 cannot authenticate K3 because there is no certificate chain from K7 to K3. However, in the composed certification graph in Figure 6.6 (b), K5 and K8 are certified by the CA and therefore trusted. K7 in the composed system can authenticate K3 by following a chain from (K5→K4→K3). The confidence values of certificate chains also increase due to the distributed CA. For example, the confidence value that K2 has for K5 is $0.7 * 0.9 * (1 - 0.1)^1 = 0.567$ (with $p = 0.1$ for attenuation) in the original certification graph using the chain (K2→K3→K5). In the composed graph, the authentication has an increased confidence value of 0.95 following a direct chain from the CA (CA→K5). Such composition is simple and cost-effective and can enhance any certificate chaining system. We are currently studying the effect of this configuration on real-world certificate chaining systems like PGP [Zim95].

6.3 Evaluation

We demonstrate the effectiveness of Composite Key Management through two sets of experiments. We simulate stressful but realistic scenarios for a distributed CA or for certificate chaining and the effect of introducing Composite Key Management. We first generate a set of relatively sparse certification graphs using pure certificate chaining. Since certificate chaining cannot provide an adequate authentication service with these sparse graphs, we certify some fraction of the nodes in the network using a distributed CA. These CA-certified nodes produce more certificate chains in the graph, improving the overall success ratio. Also, the certified nodes enable discovery of shorter and more trustworthy chains due to using the distributed CA as a trust anchor, improving the overall quality of authentication. When measuring the quality of authentication, we present the results from using a single chain with the highest QoA value through this chapter. Since the goal of the evaluation in this chapter is to show the benefit of composition over any existing mechanisms, it is best to consider each authentication instances (i.e., a certificate chain) separately. In the next chapter, we extend this metric to include all available certificate chains for complete comparison study.

Number of Total Nodes	150
Number of MOCA nodes	30
Crypto Threshold k	1, 2, 5, 10, 15, 25, 30
Network Area	1000m x 1000m
Simulations Time	600 seconds
Certificate Request Pattern	10 requests from 100 client nodes (Total 1000 requests)
Mobility	Max speed of 20m/s, 10 sec pause time

Table 6.1: Simulation Parameters for ns-2

6.3.1 Composing a Distributed CA with Certificate Chaining

By composing the distributed CA with certificate chaining, composite key management increases the availability and maintains strong security of the distributed CA. While Composite Key Management can be applied to any kind of distributed CA scheme, we choose our own MOCA distributed CA for this experiment. While MOCA adheres to the security component of the node participation principle, it has been shown that MOCA cannot achieve a 100% success ratio under stressful situations due to mobility and intermittent connectivity. In this experiment, 1-hop certificate chaining is used to augment the MOCA framework. Any node that has been certified by the MOCA framework can issue certificates to other nodes.

In our simulation set-up, out of 150 total nodes, 30 nodes are selected to serve as the MOCA nodes. To stress the MOCA distributed CA, we conducted two different types of simulations. We first evaluate the effect of mobility on the availability. Second, we evaluate the effect of the crypto threshold k by fixing the number of MOCA nodes in the network and increasing k. Crypto threshold is a common parameter to any distributed security service relying on a quorum of nodes to reach a decision. In this case, k is the minimum number of MOCA nodes a client must contact to receive certification service. In all simulations, when a node requests a certification service, the node first tries to contact the distributed CA. If that fails, the node probes its 1-hop neighborhood to check if there are any CA-certified nodes. All simulation results are an average of five different scenarios with the same parameters in different topologies. Simulation parameters are shown in Table 6.1.

To evaluate the effect of composing MOCA with 1-hop certificate chaining on the success ratio, we fixed the crypto threshold k to 15 and gradually increased mobility. As shown in Figure 6.7, the success ratio of MOCA degrades from 92% to 78% as mobility increases. However, the 1-

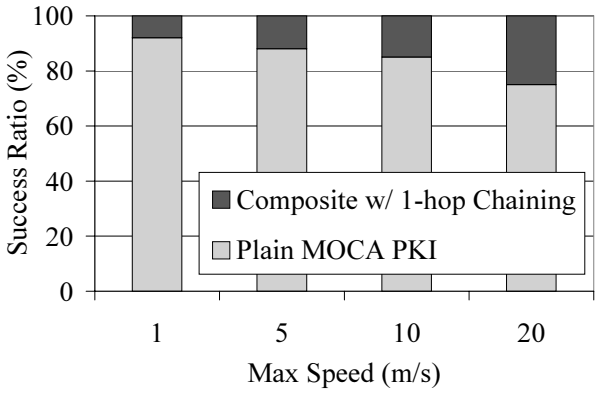


Figure 6.7: Success Ratio vs._Mobility, $k = 15$

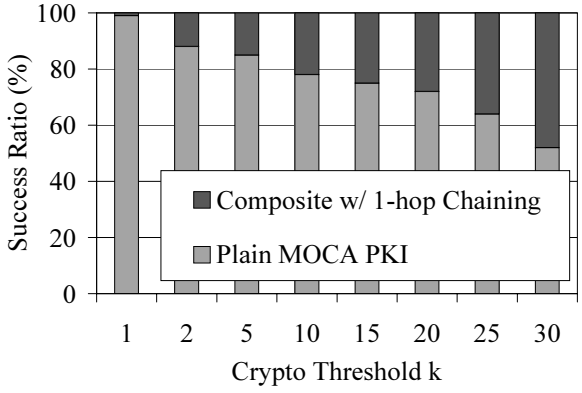


Figure 6.8: Success Ratio vs._Crypto Threshold k, max speed = 20 m/s

Max Speed (m/s)	1	5	10	20
MOCA Packet Overhead (pkts)	9234	130423	170375	190241
Chaining Packet Overhead (pkts)	189	256	332	503
Overhead Increase (%)	2	0.1	0.3	0.2

Table 6.2: Communication Overhead for Composite Approach, k=15

hop certificate chaining always succeeds in filling the gap and improving the success ratio to stay between 99.2% and 100%. Similarly, Figure 6.8 shows results from varying the crypto threshold k with a fixed maximum speed of 20m/s. When $k = 1$, a client only needs to contact one MOCA node and the success ratio is 99%. However, as k increases, a client node must contact more MOCA nodes and the success ratio decreases. In the extreme case of $k = 30$, the success ratio drops to 52%. When 1-hop chaining is added, a 99-100% success ratio is achieved. As shown from these two evaluations, even the simplest form of certificate chaining can alleviate the availability problems of a distributed CA.

The average confidence value for these experiments is affected by the number of MOCA authentications and the number of authentications using 1-hop chains. It is important to note that in both of these experiments, the average confidence value is reduced when composite key management is used. However, this decrease is the result of including the lower confidence values of the authentications using 1-hop chains and these authentications would not have been successful with MOCA alone. Additionally, the successful authentications from MOCA retain their confidence values with composite key management. Essentially, more nodes are authenticated but with lower confidence values.

The benefits of 1-hop certificate chaining comes with negligible overhead. Both communication and computation overhead have been observed to be negligible since chaining is only invoked when MOCA authentication has failed. Communication overhead is localized to 1-hop neighbors and each certificate request consists of a single broadcast request packet and one or more reply packets. The packet overhead from a simulation with fixed $k = 15$ and varying mobility is shown in Table 6.2. Results from varying simulation parameters show similar trends. Packet overhead stays under 2% of the MOCA overhead in all cases.

This composite approach satisfies both principles without sacrificing security or availability. This demonstrates that limiting the core security functionality of the CA to a small fraction of

trusted nodes can maintain high security and availability at the same time by using the remaining nodes to provide extended coverage of the distributed CA.

6.3.2 Composing Certificate Chaining with a TTP

The fundamental problems with certificate chaining stem from the fact that the certification graph is generated by the voluntary actions of individual nodes. When not enough nodes have issued certificates, the certification graph is too sparse to contain any chains between a given pair of nodes. Even when the certification graph becomes dense enough to provide certificate chains, the validity of such chains remains questionable due to the dependence on the correct behavior of the participating nodes. We evaluated the effect of composing certificate chaining with a TTP using several realistic scenarios that stress this limitation.

Since the composite approach is aimed at ad hoc environments, we generated certification graphs using a popular ad hoc mobility pattern generator `set-dest` with corrections for the speed-decay problem [YLN03]. All mobility patterns include 100 nodes moving in 5000m by 5000m area for 600 seconds in simulation time. When any two nodes stay in each other's transmission range for longer than one minute, we assume that these two nodes always issue certificates to each other. A one minute threshold is chosen to give nodes enough time to check each other's identity as well as to create and issue certificates. Such certification graphs are as dense as possible for the given mobility patterns. To simplify the evaluation, every certificate is issued with a maximum confidence value of 1.0. These two choices allow pure chaining to achieve the best possible performance, setting the baseline as high as possible for fair demonstration of improvements from composition. The injection of CA-certified nodes into the certification graph is achieved through random sampling. Nodes are randomly labeled as CA-certified up to the target fraction of CA-certified nodes. Since the simulation area is relatively large compared to the number of deployed nodes, the density is not very high, resulting in sparse certification graphs. We study the effect of varying the fraction of certified nodes and the maximum speed. Pause time in all patterns is fixed to 60 seconds.

Any variation of certificate chaining can be used for composition. However, the results from our experiment represent the best achievable results for any certificate chaining approach since all nodes are provided with complete knowledge of the full certification graph. For example, an

approach like Capkun et al. [CBH03] divides up the certification graph across multiple nodes. It is highly likely that a subset of mobile nodes can only recreate a part of the certification graph. In such cases, the resulting certification graph will be sparser than the full graph and the performance of pure chaining will degrade.

To evaluate composing certificate chaining with a TTP, we consider two metrics: the number of successful authentications and the quality of authentication. For the quality metric, we measure the average confidence value from all chains resulting in successful authentications.

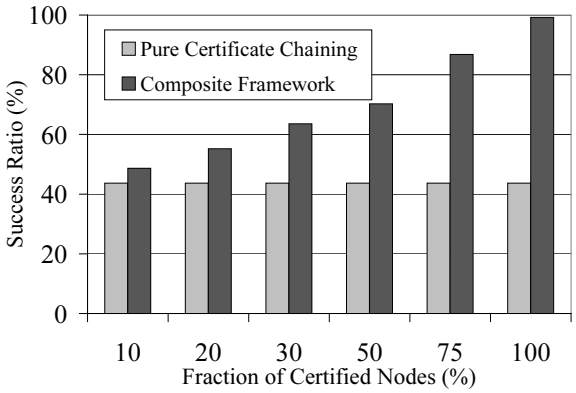


Figure 6.9: Success Ratio vs. Fraction of Certified Nodes

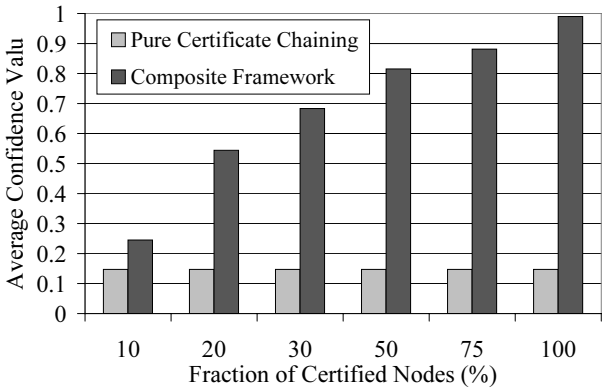


Figure 6.10: Average Confidence Value vs. Fraction of Certified Nodes

In the first set of experiments, the fraction of certified nodes is increased from 0% to 100% with the maximum speed fixed at 10m/s. With no certified nodes in the network, only 44% of all possible pairs of nodes can authenticate each other in pure certificate chaining (See Figure 6.9). As the fraction of certified nodes increases, the number of successful authentications for the composite approach increases significantly and reaches a 100% success ratio when every node in the

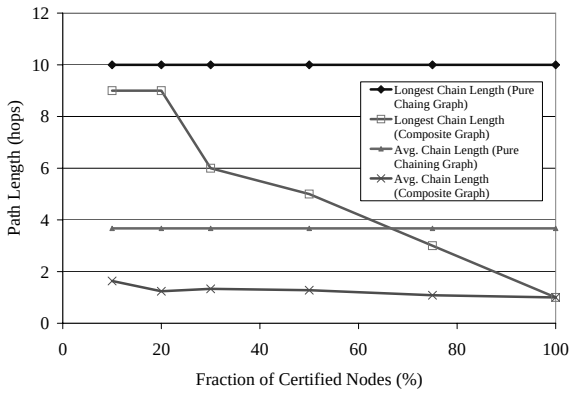


Figure 6.11: Path Lengths, with varying fraction of certified nodes

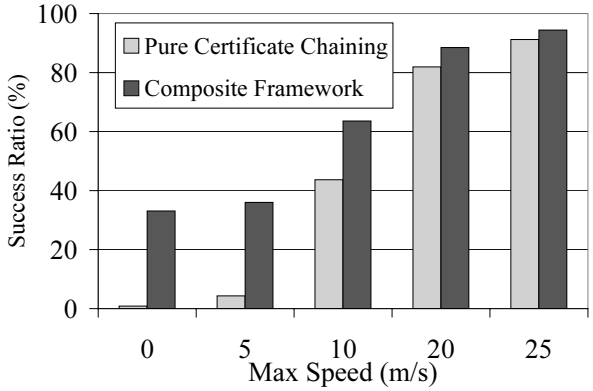


Figure 6.12: Success Ratio vs._Mobility, with 30% of certified nodes

network is certified. With 10% of the nodes certified, the composite approach provides 11% more authentications, while with all nodes certified, the improvement goes up to 128%.

Figure 6.10 presents the average confidence values from all certificate chains used for successful authentications. As clearly displayed in Figure 6.10, the average confidence value in the composite framework is 66% to 570% higher than in pure certificate chaining. This is due to fact that with many certified nodes in the network, the length of the certificate chains decreases, resulting in higher confidence values.

Figures 6.12 and 6.13 present similar results for varying mobility. As the maximum speed increases from 0 m/s to 25 m/s, the success ratio of pure certificate chaining improves from 0.8% to 94%. As nodes move faster, they travel farther and have a higher chance to meet more nodes, resulting in a denser certification graph. The success ratio of the composite approach also increases for the same reason. However, in this experiment, Figure 6.13 is more interesting. While the average

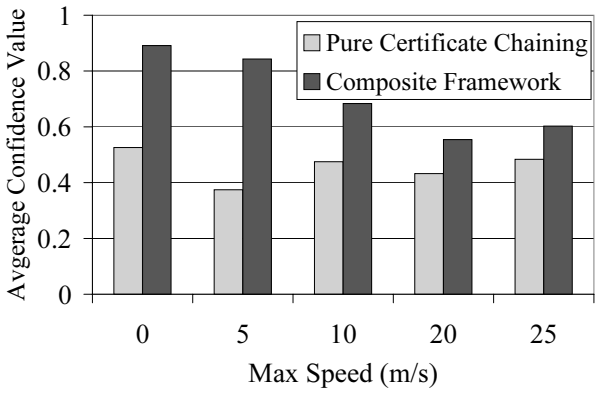
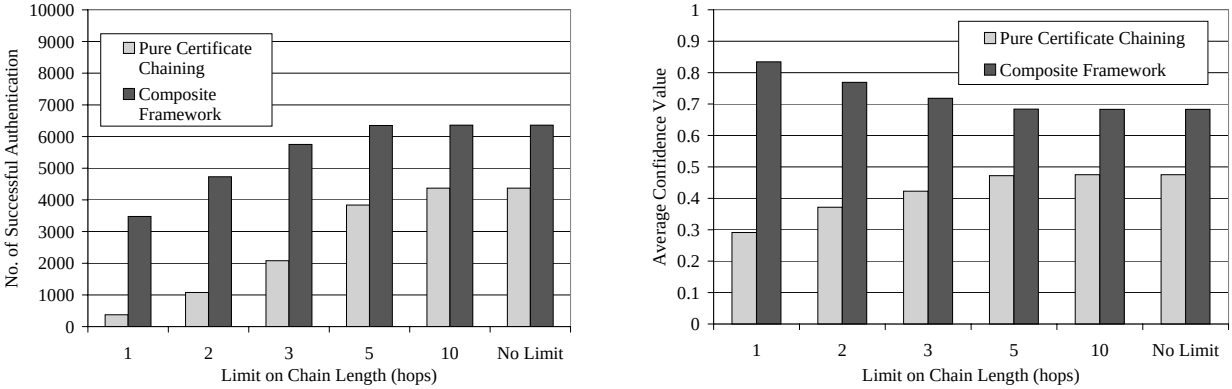


Figure 6.13: Average Confidence Value vs._Mobility, with 30% of certified nodes

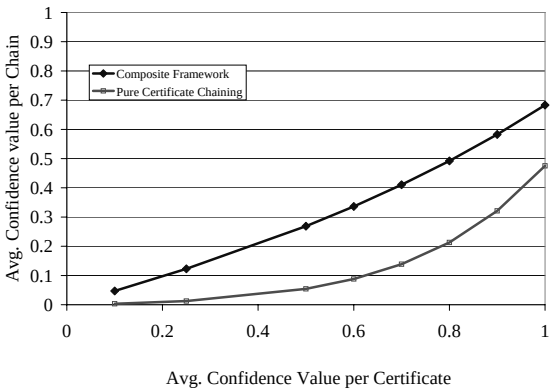


(a) No. of Successful Authentication (out of 10,000) (b) Avg. Confidence Value

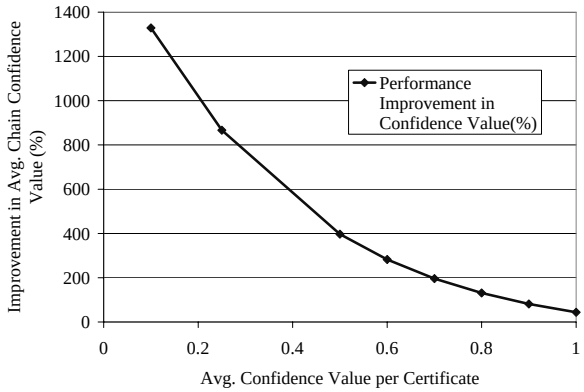
Figure 6.14: Varying Limit on Chain Length, 30% certified nodes, 10 m/s max. speed

confidence value of pure chaining stays within bounds, it gets worse in the composite approach. Originally, the confidence value from the composite approach is high since most authentications rely on CA-certified nodes. However, a denser certification graph through high mobility improves the overall success ratio by creating more chains with lower confidence values and these new chains reduce the the average confidence value.

Effect of Limits on Chain Length Reiter and Stubblebine proposed to limit the maximum allowable length of a certificate chain to avoid using questionable chains and increase the confidence of overall authentication [RS99]. To evaluate the effect of this restriction, we vary the limits on chain length. If a discovered chain is longer than the maximum allowable length, the chain is simply discarded. As the length-limit increases, the success ratio increases for both pure certificate



(a) Avg. Confidence Value of Certificate Chains



(b) Improvements in Avg. Chain Confidence Value

Figure 6.15: Varying Avg. Confidence Value, 30% certified nodes, 10 m/s max. speed

chaining and the composite approach (See Figure 6.14 (a)). However, the composite approach reaches a plateau sooner than pure certificate chaining. Pure chaining continues to increase until the length limit is 10, while the composite approach reaches its maximum with length limit 5. This indicates that the composite approach generates shorter chains than pure chaining and it is never necessary to use a chain longer than 5 in this scenario. (The actual longest chain in the composite graph has 6 hops but there are only two of them so the effect of omitting them was minimal). As shown in Figure 6.14 (b), the average confidence values of pure chaining do not change very much. However, with the composite approach, the average confidence values keep decreasing as more longer chains are introduced. This again shows the negative effect of using long chains for authentication purposes.

Effect of Average Certificate Confidence Value Finally, Figure 6.15 shows the effect of heterogeneous confidence values for individual certificates on the overall quality of authentication. As the average confidence value for individual certificates grows from 0.1 to 1.0, the average chain confidence values for both pure chaining and the composite approach increase. However, as shown in Figure 6.15 (b), the improvement from the composite approach is greater when individual certificate confidence levels are low. For example, with average of 0.1 for certificate confidence value, the composite approach provides certificate chains achieving 1328% higher confidence values. As individual certificate confidence values grow, the improvement slows down to 43% when every cer-

tificate in the network has a maximum confidence value. This shows that the composite approach can improve existing approaches even more under stressful situations.

6.4 Summary of Contributions

In this chapter, we propose a novel approach for key management in ad hoc network called *Composite Key Management*. Based on the two principles from Section 4.1.2, detailed mechanisms to implement a Composite Key Management framework are presented. Using two representative configurations of Composite Key Management, we demonstrate the effectiveness of composition. Through extensive simulation studies, we demonstrate that Composite Key Management satisfies both principles for ad hoc key management and can provide flexible, modular, and adaptive key management services for ad hoc networks. Using the same summary diagram from the previous chapter, we can clearly see that the Composite Key Management excels both in the quality of authentication and availability in Figure 6.16.

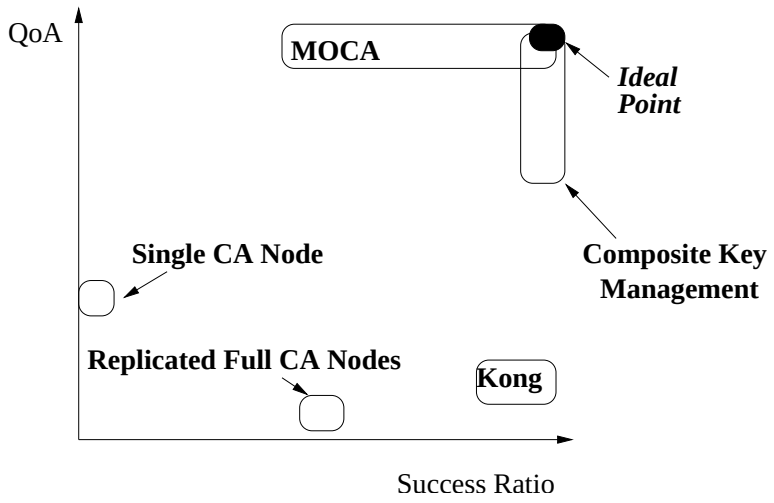


Figure 6.16: Performance Comparison of Composite Key Management

In the next chapter, we complete our study of situation-aware ad hoc key management with an extensive comparison study of all existing ad hoc key management frameworks using our QoA metric.

Chapter 7

Quality of Authentication of Ad Hoc Key Management Frameworks

In previous two chapters, we presented our solutions for ad hoc key management with novel metrics to measure the quality of authentication service they provide. In this chapter, we apply our comprehensive QoA metric to existing ad hoc key management frameworks including our own to provide a complete the study of situation-aware ad hoc key management.

Most previous research on ad hoc key management frameworks has been evaluated with two metrics: *success ratio* and *overhead*. While accurate overhead measurements can adequately express the cost of using the given key management framework, success ratio alone does not correctly show the impact of situational factors on the utility of the framework. We showed that there is a significant difference between the quality of authentication between CA-based PKIs and certificate chaining approaches in Chapter 6. Therefore, a key management framework's utility must be measured in both dimensions: *QoA* and *success ratio*. Therefore, we use our MoA to extensively study the QoA of existing key management frameworks. While the focus of this study is the measurement of QoA, we also measure the success ratio to provide a complete evaluation. Additionally, we introduce a normalized QoA metric that integrates QoA and success ratio, allowing for a comprehensive evaluation of overall performance. First, we present the metrics of interest in more detail. Second, we discuss how the experiments are set up to compare different designs of key management frameworks. Based on both analytical and simulated results, we evaluate existing key management frameworks including two distributed PKI, one distributed certificate chaining, and one hybrid approach.

7.1 Evaluation Criteria

The main focus of this study is QoA measurement. However, measuring only QoA is no better than measuring success ratio alone, as in most previous studies. Both QoA and success ratio must be measured together for comprehensive evaluation of the utility of key management frameworks.

7.1.1 Quality of Authentication

Using the QoA metric presented in Section 4.2.2, the QoA Engines in the client nodes calculate the QoA for a single authentication instance. To evaluate the quality of service provided by a key management framework, we use the average QoA value over all successful authentication instances, which separates out the effect of unsuccessful authentications.

$$QoA(framework) = \frac{\sum QoA(Authentication\ Instance)}{No.\ of\ Successful\ Authentications}. \tag{7.1}$$

In a distributed PKI approach, the QoA of the framework is simply the security level of the distributed CA since all authentications are performed through the distributed CA. Therefore, the QoA of a distributed PKI approach is calculated analytically based on the configuration parameters from Equation 4.1. In a certificate chaining approach, the QoA of each authentication depends on the shape and contents of the local TRG of the authenticating node, which depends on the operating conditions of the network. Therefore, the QoA of the framework can only be measured through extensive simulation by calculating the QoA for all possible instances under varying conditions. For hybrid approaches, both analytical measurements for the distributed PKI component and simulation studies for the certificate chaining component must be measured and combined.

7.1.2 Success Ratio

Along with QoA, success ratio is another key metric to evaluate the utility of a key management framework. Success Ratio is defined as:

$$Success\ Ratio = \frac{No.\ of\ Successful\ Authentication}{No.\ of\ All\ Authentication\ Attempts}. \tag{7.2}$$

In a distributed CA approach, a successful authentication is achieved when the authenticating

node successfully contacts the distributed CA and receives a certification service. In other words, success ratio for a distributed CA is defined by the the availability of the CA, which depends on mobility, topology, and CA access patterns in the network. We use the ns-2 network simulator to measure the success ratio of distributed PKI approaches under various scenarios.

For a certificate chaining approach, the success of authentication solely depends on the authenticating node's local knowledge about trust relationships in the network. In other words, the shape and contents of the authenticating node's local TRG determine the success of an authentication attempt. The shape of a local TRG depends on two factors: the shape of the global TRG and the TRG distribution mechanism. If the global TRG does not contain any chain linking the authenticating node and the target node, authentication always fails. Even when there is a certificate chain in the global TRG, an authentication can still fail if the local TRG of the authenticating node does not contain the relevant chain. Therefore, success ratios for a distributed certificate chaining approach measure the effectiveness of the TRG distribution mechanism. We generate a suite of global TRGs and their matching set of local TRGs from mobility patterns generated from the ns-2 network simulator.

In this way, it is possible to enforce the same type of mobility patterns and network conditions on both distributed PKI and certificate chaining approaches. For a hybrid approach, a successful authentication can be provided with either the distributed PKI or the certificate chaining component. We showed that it is desirable to use distributed PKI when possible, then use the certificate chaining as the back-up measure, to maintain a high quality of authentication in Chapter 6. Following this intuition, we designed our experiments to force the client nodes to first try to use the distributed PKI service and fall back to certificate chaining only when PKI cannot provide service.

7.1.3 Normalized QoA

For a more intuitive comparison, we can combine the QoA of a framework and the success ratio into a single metric called *Normalized QoA* defined as:

$$Normalized\ QoA = QoA * Success\ Ratio. \tag{7.3}$$

Normalized QoA can be understood as the average QoA of all authentication instances including

both successful and unsuccessful. Since normalized QoA includes success ratio, it can only be calculated for a set of *past* authentication results and not per authentication basis. However, normalized QoA embodies the idea of situation-aware security by integrating all situational impacts on authentication services and allows the network operators to understand the overall performance in a comprehensive manner.

7.2 Experiment Set-Up

To evaluate and compare different approaches, we use the ns-2 network simulator and its mobility generator. The same set of simulation parameters, including the number of mobile nodes, the operating area of the network, the mobility patterns of nodes, the authentication request patterns, and simulation time are enforced between different approaches for fair comparison. In all simulations, the test network contains 150 mobile nodes. Two different simulation areas, of sizes 1500m x 500m and 3000m x 3000m, are used. In all cases, the network runs for 500 seconds in simulation time before measurement begins. This warm-up phase is to eliminate the effects of a network cold start and also to give enough time for certificate chaining to *"boot up"* so that TRGs can reach a stable state. The rest of the simulation parameters are listed in Table 7.1.

Total Number of Mobile Nodes	150
Area of Network	1500m x 500m, 3000m x 3000m
Total Simulation Time	1000 seconds (including 500 sec. warm-up time)
Node Pause Time	10 sec.
Maximum Node Speed	0, 1, 5, 10, 15, 20, 30, 50 m/s

Table 7.1: Simulation Parameters

7.3 Experiments

7.3.1 Distributed PKI Approaches

Since all successful authentications in a PKI approach are through the certificate authority and no other nodes, measurement the QoA of distributed PKI approaches can be acquired by the analytical calculation of the CA security level as presented in Section 4.2.2. The QoA Engine in a client node

can calculate a distributed CA's security level based on its threshold cryptography configuration. Since the CCV for a CA-based authentication instance relies solely on the security level of the CA for the confidence value as defined in Equation 4.3, the average QoA for a distributed PKI approach is the average of the CA's security level observed by all end users in the system. In our simulation study, we assume that all end users share the same perception of the network status and have a single c value. The CA's security level presented in Equation 4.1 for the experimental settings behaves as in Figure 6.2.2, which is based on the calculation with the example crypto threshold k = 10. Given a fixed k, varying c or more specifically varying the *gap* between k and c affects the CA's security level.

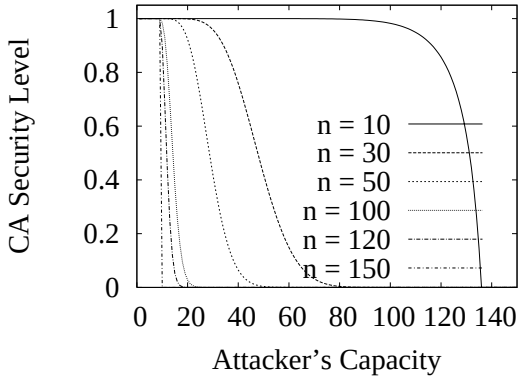


Figure 7.1: CA Security Level

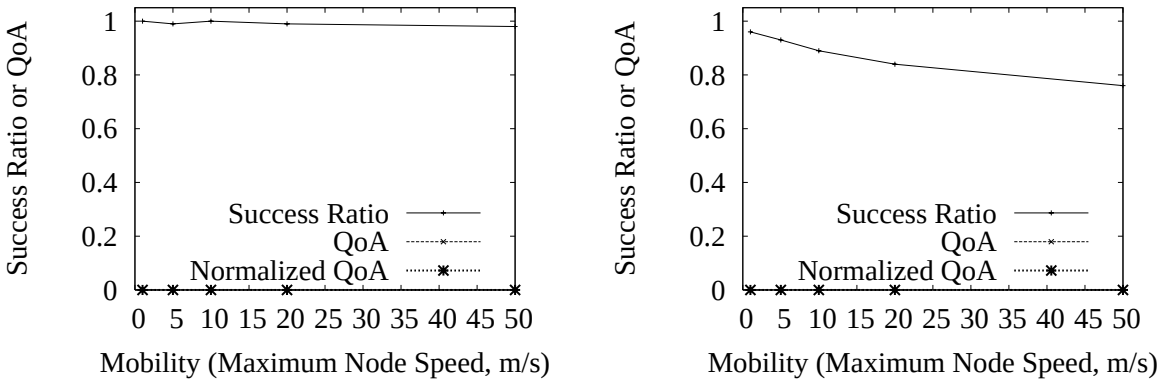
In Figure 6.2.2, each curve shows different configurations of a distributed PKI approach with varying number of CA nodes. There are two important properties to note. First, each curve has a sharp drop point where the CA's security level drops from a very high value close to 1.0 to a very low value close to 0.0. This means that the security of a distributed CA based on threshold cryptography does not degrade gracefully as attackers become more capable, emphasizing the importance of picking the right number of CA nodes depending on the operating conditions. Second, the curves move to the left as the number of CA nodes increases, showing that it is important to limit the number of CA nodes to a small fraction. The leftmost line shows the case for Kong's approach (i.e., $n = 150$) where the security level drops from 1.0 to 0.0 as soon as the attacker becomes capable of compromising k nodes.

Kong et al.'s Distributed CA

The success ratios for distributed PKI approaches are measured with ns-2 network simulations where the authenticating nodes try to contact the distributed CA when they need to authenticate another node. Since testing all possible instances of authentication in 150-node networks requires excessive amounts of time to simulate (22,500 certification requests must be served), we use a random sampling to reduce the amount of simulations to be performed. More specifically, 100 nodes are randomly selected and they each choose 50 other random nodes and attempt to authenticate each of them in turn. This test covers 5,000 authentication instances (around 22% of all cases). We run five different mobility scenarios for each set of simulation parameters and the results presented here are the average from the five simulation runs. All results presented in this section are from scenarios with attacker's capacity $c = 10$ and crypto threshold $k = 10$. If $c < k$, the CA is protected by the threshold cryptography. An adversary with a capacity $c = k$ is the least powerful attacker that can compromise the CA. We assume that there is one such least powerful attacker in evaluation of Kong's scheme because Kong's scheme becomes completely vulnerable as long as $c \geq k$ and increasing the attacker's capacity beyond k (i.e., $c > k$) does not have any effect on the CA's security level.

Kong's distributed PKI uses all mobile nodes in the network as distributed CA nodes. Every node shares a piece of the CA's secret key and can participate in providing authentication service. In our experiments, the total number of mobile nodes is fixed to 150, as is the number of CA nodes, by design. Various crypto threshold values, $k = 1$, 5, 10, 20, 30, have been tested. Intuitively, a higher k value should decrease the success ratio and improve the security of the distributed CA. However, this effect is not seen in Kong's design since (1) success ratio is always maintained very high due to their design decisions to use all mobile nodes as CA nodes, and (2) the security level of Kong's CA is constantly low as long as the attacker's capacity, c, exceeds the crypto threshold k.

Figures 7.2 (a) and (b) each show QoA, success ratio, and normalized QoA for 1500m x 500m and 3000m x 3000m test area. Even though the test area in the second case is considerably larger, resulting in a much lower node density, the success ratio of Kong's distributed CA still remains high. This is due to the high redundancy of CA nodes. However, the same design choice makes it very easy to compromise the distributed CA as long as the attacker is powerful enough. The



(a) 1500m x 500m area (b) 3000m x 3000m area

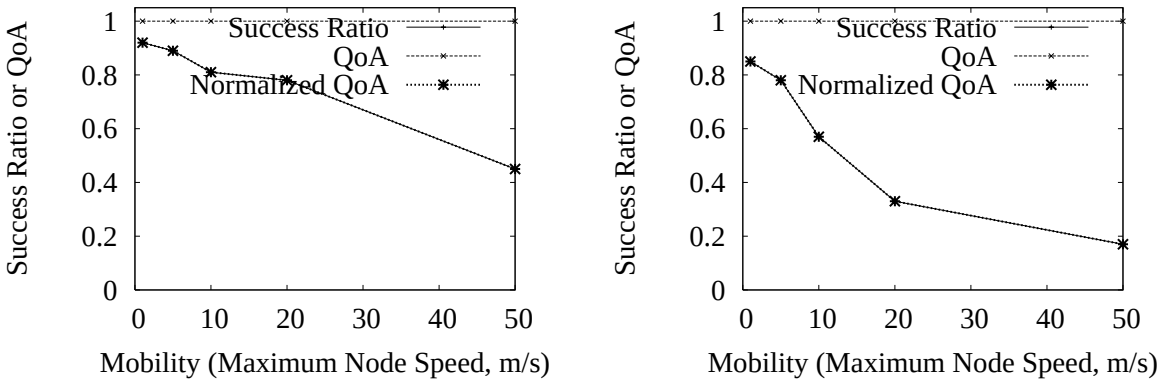
Figure 7.2: Kong's Distributed PKI

attacker can always compromise Kong's framework, which yields the CA security level close to zero, resulting in very low QoA values. As a result, the normalized QoA is heavily affected by the low QoA and also remains very low. In summary, Kong's approach is an example where the quality of security is sacrificed to provide perfect availability. This design decision is captured by applying the situational information and can now be exposed to end users.

MOCA

We presented the MOCA distributed PKI in Chapter 5. They enhance the security of the distributed CA by limiting the number of CA nodes to a relatively small fraction while hiding their identities so that attackers cannot focus their attack resources on the CA nodes. Also, the CA nodes in MOCA are selected based on their node characteristics so that they can resist potential attacks. In our experiments, we tested with varying numbers of CAs, between 15 and 50, which are respectively 10% and 33% of the total mobile nodes. In this section, we present a case with $n = 30$ where 20% of the mobile nodes in the network are selected as distributed CA nodes. Results presented here use the fixed values of crypto threshold $k = 10$, and the attacker's capacity $c = 10$ for a fair comparison with Kong's scheme. In MOCA, as the attacker grows stronger (i.e., with larger c), the CA's security level stays high much longer and then degrades gracefully compared to Kong's scheme.

In both Figures 7.3 (a) and (b), the security of the MOCA framework is maintained high, very



(a) 1500m x 500m area (b) 3000m x 3000m area

Figure 7.3: MOCA Distributed PKI

close to 1.0. This shows the effect of the design choices to enhance the security of the distributed CA. However, the price for maintaining such a high level of security is shown in the success ratio in both graphs. In Figure 7.3 (a), the success ratio gradually degrades from 92% to 45% as the mobility in the network increases. While 45% success ratio under the extreme mobility of 50 m/s does not seem too bad, the effect on success ratio is more amplified in Figure 7.3 (b), where the node density is considerably lower than Figure 7.3 (a), which causes more difficulty to contact the required number of distributed CA nodes. The success ratio in Figure 7.3 (b) degrades from 85% to 17%. In the MOCA framework, the success ratio dominates the normalized QoA since it provides a very high QoA constantly with varying success ratios. MOCA is an example where the authors first achieve very strong security of the framework and provide the best-achievable communication support for the given availability, demonstrating the applicability of the *Situation-Aware Security* paradigm for ad hoc security service design.

7.3.2 Certificate Chaining Approaches

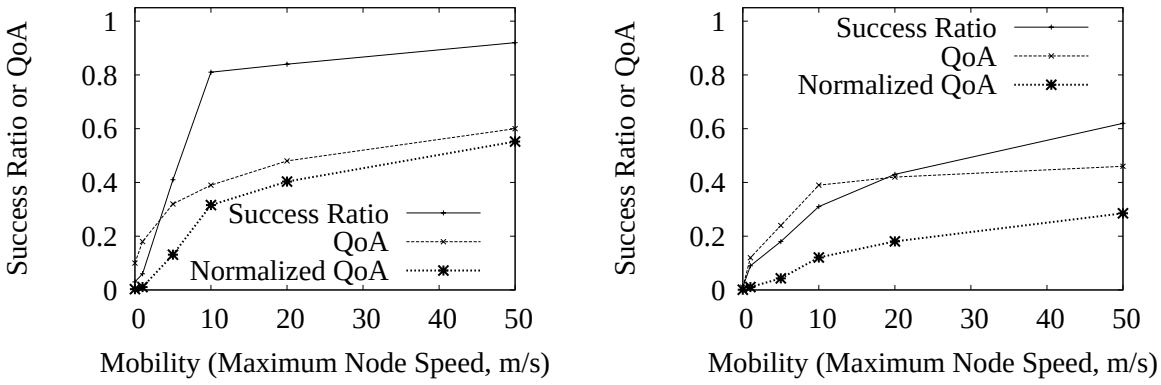
Measurement of QoA for certificate chaining is based on the contents of the local TRG of authenticating nodes at the time of the authentication attempt. If the authenticating node has more information (or a larger portion of the global TRG), both QoA and the success ratio improve. The success ratio of certificate chaining again depends on the snapshot of the local TRG of the authenticating node at the time of the authentication attempt. Since the best achievable QoA and the

success ratio of distributed certificate chaining are limited by the shape of the global TRG, we first simulate the ideal case where every mobile node has perfect knowledge of the trust relationships in the network and compare the results with a distributed certificate chaining mechanism proposed by Capkun et al. [CBH03].

To compare certificate chaining with distributed PKI approaches, we enforced the same mobility patterns and network settings for both simulations. Based on the mobility log from each 1000-second simulation run, we populate the global trust relationship graph by adding an edge (or issuing a certificate) when a node is in communication range of another node for more than one minute. This is to simulate the time to identify each other, collect relevant data from each other, and issue certificates with a full confidence value of 1.0 to each other. This results in best-case scenarios for certificate chaining where every trust relationship is bidirectional (which is not always true in reality) and has the maximum level of confidence. In a way, we are creating the best-possible scenarios for certificate chaining that generate the upper-bound for QoA and success ratio. We show next that even with these strong assumptions, certificate chaining can only provide marginal performance due to its inherent dependence on transitive trust. For the results in the graph, we use $\gamma = 0.3$ for the impact factor when combining multiple chains.

Using the Global Trust Relationship Graph

The first set of results for certificate chaining is from the ideal case where every mobile node has perfect knowledge about the global TRG. After generating the global TRG during the 1000-second simulation run, authentications between every pair of nodes is performed using the global TRG. Both QoA and success ratio improve in Figures 7.4 (a) and (b) as mobility increases. This is due to the increase number of nodes that a mobile node encounters as they move faster. With 30 second pause time, two mobile nodes that encounter each other issue certificates to each other with a high probability, thus populating the TRG. Since a denser TRG contains more certificates chains, both QoA and success ratio are improved. Unlike the cases with distributed PKI approaches where either success ratio or QoA dominates the effect on normalized QoA, the normalized QoA of certificate chaining is affected by both. Normalized QoA also improves as mobility grows due to the improvement in both success ratio and QoA.

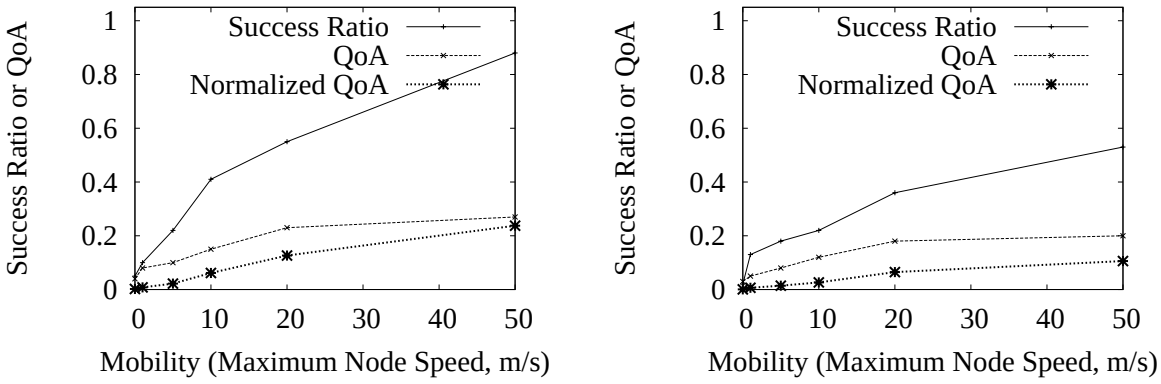


(a) 1500m x 500m area (b) 3000m x 3000m area

Figure 7.4: Certificate Chaining with Global Trust Relationship Graph

Distributed Certificate Chaining by Capkun et al.

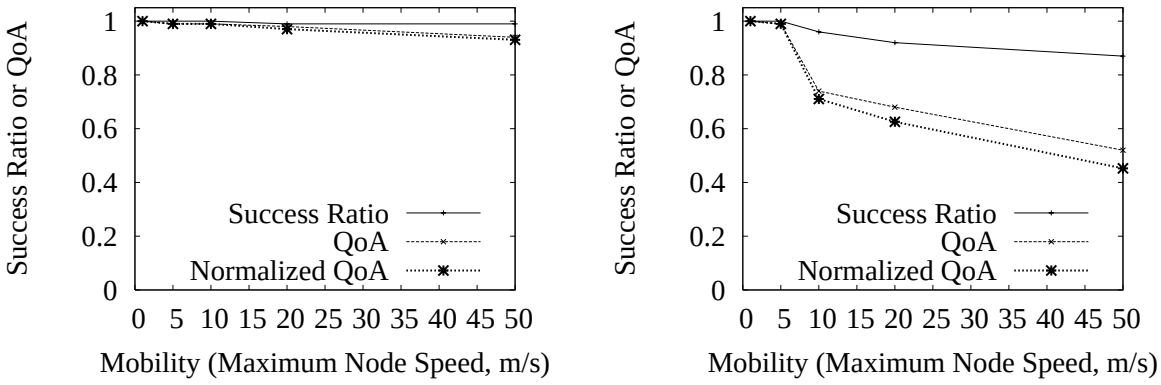
Capkun et al. proposed a distributed version of certificate chaining without having a centralized component that stores the global TRG [CBH03]. Basically, each mobile node carries a subgraph of the global TRG and nodes enhance their knowledge as they encounter more nodes. We use the same method as in the global TRG test where each mobile node generates their local TRG during the 1000-second simulations runs according to the scheme proposed in the paper. Then, each node attempts to authenticate every other node in the network only using the local TRG. As shown in



(a) 1500m x 500m area (b) 3000m x 3000m area

Figure 7.5: Distributed Certificate Chaining by Capkun et al.

Figure 7.5 (a), Capkun's distributed version provides a comparable success ratio to using the global TRG in all ranges as shown in their original paper. However, since each mobile node has less than



(a) 1500m x 500m area (b) 3000m x 3000m area

Figure 7.6: Composite Key Management

perfect knowledge about the global TRG, many authentication instances based on the local TRG use longer certificate chains, resulting in the lower overall QoA values. This effect is even more clearly amplified in Figure 7.5 (b) where the node density is much lower and the mobile nodes are challenged to gather enough information about the global trust relationships. The success ratio in Figure 7.5 (b) also shows the effect of imperfect knowledge caused by the distribution of the TRG.

Normalized QoA shows this degradation of performance in an amplified view since the effect from imperfect knowledge is represented twice through both success ratio and QoA. These results again emphasize the importance of *situation-aware security* where the quality of security is constantly monitored since a naive comparison of success ratios alone does not clearly demonstrate the difference between authentications based on global knowledge and partial knowledge.

7.3.3 Hybrid Approaches

We presented the performance of our Composite Key Management framework in Chapter 6 that it can achieve both high success ratio and high QoA using the single chain CCV. In this section, we present results from utilizing all available node-disjoint certificate chains. When there are two mechanisms in place, an end user can choose freely from either one. However, in this experiment, we force the mobile nodes to first try the distributed PKI and then fall back to certificate chaining when PKI cannot provide authentication service. This is following their original experiment where the use of PKI is preferred due to its higher QoA support.

As shown in Figure 7.6 (a), Composite Key Management achieves a very high success ratio while maintaining a very high QoA. However, under more challenging scenarios in Figure 7.6 (b), while success ratio is still maintained at relatively high levels, the quality of authentication starts to degrade as the distributed CA becomes unavailable and mobile nodes become challenged to maintain their local TRG. Normalized QoA for the first set stays very close 1.0, showing almost perfect quality of security. However, under more challenging conditions, normalized QoA drops down to 0.45, showing that even this hybrid approach, which is specifically designed for ad hoc environments, cannot always provide perfect security. This observation again emphasizes the need for situation-awareness where the fluctuations in the quality of provided security is constantly monitored and conveyed to the end users.

7.4 Summary of Findings

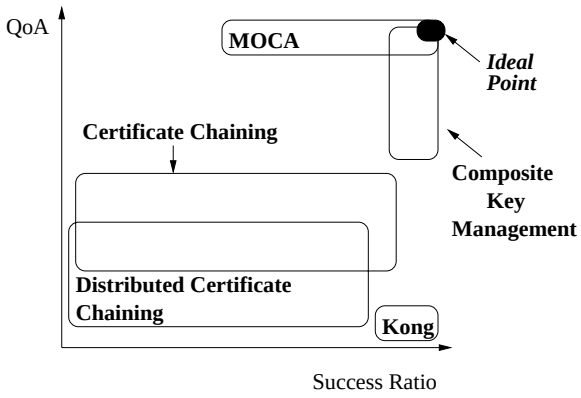


Figure 7.7: Performance Summary of Ad Hoc Key Management Frameworks

Figure 7.7 summarizes the various ad hoc key management frameworks evaluated in this study using a two dimensional chart. In general, it is better to be close to the ideal performance point represented as the black rectangle. Also, a smaller range, both in success ratio and QoA, means more constant results, while a larger range represents more variance in the quality of the provided service. It is clear that previous approaches that only evaluate success ratio are not enough to effectively compare different designs. Success ratios from all examined frameworks overlap each other and it is unclear how to compare their performance based on success ratio only. While Kong's approach stands out with the highest success ratio under all conditions, QoA measurements reveal the low

QoA it provides. The performance of certificate chaining and distributed certificate chaining varies over a large range of success ratios and QoA, which suggests that they might not be a good choice for scenarios that require more predictable authentication services. MOCA provides constantly high QoA with varying success ratio, which can be a good solution for environments where the quality of authentication matters more than high availability of the authentication service. Composite Key Management provides the best performance in both success ratio and QoA and the small coverage area shows a more consistent level of service. However, even Composite Key Management cannot provide perfect service under challenging conditions as shown in the previous section and this again demonstrates the importance of the *situation-aware security* paradigm in ad hoc networks. Efforts to measure the changing quality of security under varying conditions and conveying the results to end users in an intuitive manner must be parallel to the efforts to designing better-suited security services for ad hoc environments. Also, based on this comparison, network operators can choose an appropriate key management framework to deploy according to their needs. QoA completes the comprehensive evaluation of ad hoc key management frameworks and can be used as an effective guide for future designs.

7.5 Summary of Contributions

In this chapter, we extended the QoA metric presented in the previous chapter with consideration for multiple certificate chains. Following Reiter and Stubblebine's suggestion, we enforce the chains to be node-disjoin from one another and combine the CCVs from each chain using a method based on scientific confirmation theory. Extensive comparison study using our QoA metric shows that various key management frameworks provide drastically different QoA, which suggests that any effective security service for ad hoc environments must be designed around the quality of the provided service based on situational information.

The importance of *situation-aware security* in ad hoc authentication services is clearly demonstrated in this chapter since none of existing key management framework designs are immune to the environmental effects from ad hoc environments. Even the Composite Key Management designed specifically for the ad hoc environments does not provide perfect availability under challenging network conditions. Therefore, it is imperative to augment ad hoc key management frameworks with

situation-awareness component and convey the continuous QoA measurement back to end users so that they can make well-informed decisions. There are also some inherently best-effort security services designed for wired networks, such as PGP, which can easily be augmented with support for *situation-awareness*. We plan to investigate the applicability of *situation-aware security* to these services and measure its impact.

Chapter 8

Conclusions and Future Work

Security for ad hoc environments must be viewed and understood in a very different manner due to various and significant challenges from the characteristics of ad hoc networks. Instead of focusing on providing absolutely guaranteed security services as expected in traditional environments, security in ad hoc networks must be treated as a best-effort service whose quality can change frequently and to a significant extent. To understand these complicated interactions between ad hoc environments and security services, we propose a concept of *Situation* that captures the assumptions an ad hoc security service requires and the way such assumptions are affected by environmental factors. By incorporating these interactions into *measuring* the changing quality of security service, we provide a more viable and practical approach to enable necessary security supports for ad hoc networks, thus enabling its true potential as instantaneously deployable networks. Following this paradigm, we presented two types of *situation-aware* security services: security-aware routing and key management. Security-aware routing is a novel approach to quantify the quality of protection for the data path and steer routing decisions based on the security of participating nodes. We provide two designs of situation-aware key management frameworks: MOCA distributed PKI and Composite Key Management. To complete the design, we also presented a comprehensive metric to measure the quality of authentication provided by key management frameworks.

Situation-aware security paradigm can be and should be used to design other types of ad hoc security services including secure routing, incentive systems and even positioning systems. Each type of security service must be examined to discover the respective *situations* for each service and metrics to measure the quality of security must be designed based on the situational knowledge. One promising venue of research is to automate this process. Automated discovery of situational

information of a security service and design of appropriate metrics could vastly accelerate the adoption of the situation-aware security paradigm in research community.

There are other potential venues where the situation-aware paradigm may be applied. Any decentralized, distributed system can be a good candidate including fast emerging peer-to-peer systems. While a typical peer-to-peer system enjoys more luxuries such as stable connectivity compared to ad hoc environments, the situation-aware paradigm can still help design more efficient and low-cost security mechanisms for peer-to-peer environments.

References

[AG00] N. Asokan and P. Ginzboorg. Key agreement in ad hoc networks. *Computer Communications*, 23:1627–1637, 2000.

[AHNRR02] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An on-demand secure routing protocol resilient to byzantine failures. In *Proceedings of the ACM workshop on Wireless security*, 2002.

[AKW96] A. Aggarwal, J. Kleinberg, and D. Williamson. Node-disjoint paths on the mesh, and a new trade-off in VLSI layout. In *Proceedings of 28th ACM Symposium on Theory of Computing (STOC 96)*, 1996.

[All] Wi-Fi Alliance. Wireless protected access. Wi-Fi Consortium homepage available at http://www.wi-fi.org.

[AM05] J. Al-Mutahdi. *An Intelligent Security Infrastructure for Ubiquitous Computing Environments*. PhD thesis, University of Illinois at Urbana-Champaign, 2005.

[AMCK+02] Jalal Al-Muhtadi, Roy Campbell, Apu Kapadia, M. Dennis Mickunas, and Seung Yi. Routing through the mist: Privacy preserving communications in ubiquitous computing environments. In *International Conference of Distributed Computing Systems (ICDCS 2002)*, July 2002.

[BB02] S. Buchegger and J.-Y. Le Boudec. Cooperation of nodes: The CONFIDANT approach - abstract. *ACM Mobile Computing and Communications Review (MC2R)*, 6(4), 2002.

[BBK94] T. Beth, M. Borcherding, and B. Klein. Valuation of trust in open networks. In *Proceedings of the Conference on Computer Security*, 1994.

[BH03] L. Buttyan and J. P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications*, 8(5), October 2003.

[BLNS86] A. D. Birrel, B. W. Lampson, R. M. Needham, and M. D. Scheroeder. A global authentication service without global trust. In *Proceedings of the 1986 IEEE Symposium on Security and Privacy*, 1986.

[BMJ+98] J. Broch, D. A. Maltz, D. B. Johnson, Yih-Chun Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, October 1998.

[BP75] D. Bell and L. La Padula. Secure computer system: Unified exposition and multics interpretation. Technical Report ESD-TR-75-306, The MITRE Corporation, Bedford, MA, 1975.

[BSAK95] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz. Improving tci/ip performance over wireless networks. In *MobiCom '95: Proceedings of the 1st annual international conference on Mobile computing and networking*, pages 2–11, New York, NY, USA, 1995. ACM Press.

[BSSW02] D. Balfanz, D. Smetters, P. Stewart, and H. Wong. Talking to strangers: Authentication in ad hoc wireless networks. In *ISOC Symposium on Network and Distributed Systems Security (NDSS 02)*, February 2002.

[C. 94] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[C. 99] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. In *The Second IEEE Workshop on Mobile Computing Systems and Applications*, February 1999.

[C-K97] C-K Toh. Associativity-Based Routing for Ad-Hoc Mobile Networks. *Wireless Personal Communications*, 4(2), March 1997.

[Car50] R. Carnap. *Logical Foundations of Probability.* University of Chicago Press, 1950.

[CBH02] S. Capkun, L. Buttyan, and J.-P. Hubaux. Small worlds in security systems: an analysis of the pgp certificate graph. In *Proceedings of the 2002 workshop on New security paradigms*, pages 28–35. ACM Press, 2002.

[CBH03] S. Capkun, L. Buttyan, and J. P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, page 17, 2003.

[CH03] S. Capkun and J.-P. Hubaux. Biss: building secure routing out of an incomplete set of security associations. In *Proceedings of the 2003 ACM WiSe Workshop*, pages 21–29, New York, NY, USA, 2003. ACM Press.

[CHB03] S. Capkun, J.-P. Hubaux, and L. Buttyan. Mobility helps security in ad hoc networks. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*, June 2003.

[Chu02] A. L. Chu. SPF: Security performance flexibility framework for trusted operating systems. Master's Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2002.

[CLM+99] R. H. Campbell, Z. Liu, M. D. Mickunas, P. Naldurg, and S. Yi. Seraphim: Dynamic interoperable security architecture for active networks. Technical report, Champaign, IL, USA, 1999.

[CLR+02] T. Courtney, J. Lyons, H. V. Ramasamy, W. H. Sanders, M. Seri, M. Atighetchi, P. Rubel, C. Jones, F. Webber, P. Pal, R. Watro, M. Cukier, , and J. Gossett. Providing intrusion tolerance with ITUA. In *Proceedings of the DSN-2002)*, 2002.

[CYRK03] C. Carter, S. Yi, P. Ratanchandani, and R. Kravets. Manycast: Exploring the space between anycast and multicast in ad hoc networks. In *Proceedings of the Ninth ACM Annual International Conference on Mobile Computing and Networking (Mobicom 03)*, September 2003.

[Dij59] E.W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1959.

[DMS04] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[Dou00] J. Douceur. The sybil attack. In *Proceedings of IPTPS 02 Workshop*, March 2000.

[E.] E. Crawley and R. Nair and B. Rajagopalanand and H. Sandick. A Framework for QoS-based Routing in the Internet. RFC 2386, August 1998.

[E. 99] E. M. Royer and C-K Toh. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. *IEEE Personal Communications*, April 1999.

[Ent] Entrust. Entrust, Inc. Company homepage available at http://www.entrust.com/.

[F. 99] F. Stajano and R. Anderson. The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. In *The 7th International Workshop on Security Protocols*, April 1999.

[FD92] Y. Frankel and Y. G. Desmedt. Parallel Reliable Threshold Multisignature. Technical Report TR-92-04-02, Univ. of Wisconsin–Milwaukee, 1992.

[HBC01] J. P. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proceedings of the 2nd ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001)*, 2001.

[HJP02] Y.-C. Hu, D. B. Johnson, and A. Perrig. Secure efficient distance vector routing in mobile wireless ad hoc networks. In *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '02)*, June 2002.

[HPJ02] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proceedings of the Eighth ACM International Conference on Mobile Computing and Networking (Mobicom 2002)*, September 2002.

[HPJ03a] Y.-C. Hu, A. Perrig, and D. B. Johnson. Packet leashes: A defense against wormhole attacks in wireless networks. In *Proceedings of IEEE Infocom 2003*, April 2003.

[HPJ03b] Y.-C. Hu, A. Perrig, and D. B. Johnson. Rushing attacks and defense in wireless ad hoc network routing protocols. In *Proceedings of the the Second ACM workshop on Wireless security (WiSe 03)*, 2003.

[HV02] G. Holland and N. Vaidya. Analysis of tcp performance over mobile ad hoc networks. *Wireless Network*, 8(2/3):275–288, 2002.

[J.] J. Broch and D. B. Johnson. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. IETF Internet Draft, October 1999.

[J. 97] J. Howard. *An Analysis Of Security Incidents On The Internet 1989 - 1995*. PhD thesis, Doctor of Philosophy in Engineering and Public Policy, Carnegie Mellon University, April 1997.

[Jos99] A. Josang. An algebra for assessing trust in certification chains. In *ISOC Network and Distributed System Security Symposium '99*, February 1999.

[JRN] Q. Jiang, D. S. Reeves, and P. Ning. Improving robutness of pgp keyrings by conflict detection.

[JVZ00] S. Jiang, N. Vaidya, and W. Zhao. Routing in packet radio networks to prevent traffic analysis. In *Proceedings of the IEEE Information Assurance and Security Workshop*, July 2000.

[K. 02] K. Sanzgiri and B. Dahill and B. Levine and C. Shields and E. Belding-Royer. A Secure Routing Protocol for Ad Hoc Networks. In *Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP 2002)*, November 2002.

[Ken93] S. T. Kent. Internet privacy enhanced email. *Communications of ACM*, (8):48–60, August 1993.

[KP] S. Kent and T. Polk. Public-key infrastructure (x.509) (pkix) charter. Available at http://www.ietf.org/html.charters/pkix-charter.html.

[KZL+01] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad-hoc networks. In *Proceedings of the 9th IEEE International Conference on Network Protocols (ICNP 2001)*, 2001.

[LA98] R. Levien and A. Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th USENIX Security Symposium*, January 1998.

[Lin05] Stefan Lindskog. *Modeling and Tuning Security from a Quality of Service Perspective.* PhD thesis, Department of Computer Science and Engineering, Chalmers University of Technology, Gteborg, Sweden, 2005.

[LK00] R. Ludwig and R. H. Katz. The eifel algorithm: making tcp robust against spurious retransmissions. *SIGCOMM Comput. Commun. Rev.*, 30(1):30–36, 2000.

[LSHJ04] S. Lindskog, J. Strandbergh, M. Hackman, and E. Jonsson. A content-independent scalable encryption model. In *Proceedings of the 2004 International Conference on Computational Science and its Applications (ICCSA 2004)*, May 2004.

[Mah93] P. Maher. *Betting on Theories.* Cambridge University Press, 1993.

[Mau96] U. Maurer. Modeling a public-key infrastructure. In *Proceedings ofthe Conference on Computer Security (ESORICS 96)*, 1996.

[McB98] N. McBurnett. PGP web of trust statistics. Available at http://bcn.boulder.co.us/ neal/pgpstat/, 1998.

[MCG+01] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang. Tcp westwood: Bandwidth estimation for enhanced transport over wireless links. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 287–297, New York, NY, USA, 2001. ACM Press.

[MT03] T. Moreton and A. Twigg. Trading in trust, tokens and stamps. In *Proceedings of the Workshop on Economics of Peer-to-Peer Systems*, 2003.

[ns2] The ns-2 Network Simulator. Available at http://www.isi.edu/nsnam/ns/.

[NSSP04] J. Newsome, R. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: Analysis and defenses. In *Proceedings of IEEE International Conference on Information Processing in Sensor Networks (IPSN 04)*, April 2004.

[ONY03] C. S. Ong, K. Nahrstedt, and W. Yuan. Quality of protection for mobile multimedia applications. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME2003)*, 2003.

[P. 99] P. Sinha and R. Sivakumar and V. Bharghavan. CEDAR: a Core-Extraction Distributed Ad Hoc Routing algorithm. In *The 18th Annual Joint Conference of the IEEE Computer and Communication Societies*, March 1999.

[PBB+02] D. A. Patterson, A. Brown, P. Broadwell, G. Candea, M. Chen, J. Cutler, P. Enriquez, A. Fox, E. Kiciman, M. Merzbacher, D. Oppenheimer, N. Sastry, W. Tetzlaff, J. Traupman, and N. Treuhaft. Recovery-oriented computing (roc): Motivation, definition, techniques, and case studies. UC Berkeley Computer Science Technical Report UCB/CSD-02-1175, March 2002.

[Pos81] Jon Postel. Transmission control protocol. IETF RFC 793, September 1981.

[R. 97] R. Dube and C. D. Rais and Kuang-Yeh Wang and S. K. Tripathi. Signal stability-based adaptive routing (SSA) for ad hoc mobile networks. *IEEE Personal Communications*, February 1997.

[Riv92] R. Rivest. The MD5 message-digest algorithm. IETF RFC 1321, April 1992.

[RR98] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[RS98] M. K. Reiter and S. G. Stubblebine. Resilient authentication using path independence. *IEEE Transactions on Computers*, 1998.

[RS99] M. K. Reiter and S. G. Stubblebine. Authentication metric analyis and design. *ACM Transactions on Information and System Security*, 1999.

[S.] S. Murphy and M. Badger and B. Wellington. OSPF with Digital Signatures. RFC 2154.

[S. 98] S. Singh and M. Woo and C. S. Raghavendra. Power-Aware Routing in Mobile Ad Hoc Networks. In *The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, October 1998.

[S. 00] S. Marti and T. Giuli and K. Lai and M. Baker. Mitigating Routing Misbehavior in Mobile ad hoc networks. In *The Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, August 2000.

[Sch94] B. Schneier. Description of a new variable-length key, 64-bit block cipher. *Proceedings of Fast Software Encryption, Lecture Notes in Computer Science*, 1994.

[Sho00] V. Shoup. Practical threshold signatures. *Lecture Notes in Computer Science*, 1807, 2000.

[SMGLA96] B. Smith, S. Murthy, and J.J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocols. In *Global Internet '96*, November 1996.

[SNS88] J. G. Steiner, B. Clifford Neuman, and J.I. Schiller. Kerberos: An authentication service for open network systems. In *Winter 1988 Usenix Conference*, February 1988.

[SV98] G. Smith and D. Volpano. Secure information flow in a multi-threaded imperative language. In *Proceedings of POPL*, 1998.

[TH92] A. Tarah and C. Huitema. Associating metrics to certification paths. *Computer Security*, pages 175–189, 1992.

[Tha] Thawte. Thawte, inc. Company homepage available at http://www.thawte.com.

[V. 97] V. D. Park and M. S. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. In *The 16th Annual Joint Conference of the IEEE Computer and Communications Societies*, April 1997.

[Ver] Verisign. VeriSign, Inc. Company homepage available at http://www.verisign.com/.

[W. 76] W. Diffie and M.E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 1976.

[W. 95] W. Stallings. *Network and Internetwork Security Principles and Practice*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[Win03] M. Winslett. An introduction to automated trust establishment. In *1st International Conference on Trust Management*, May 2003.

[WMB99] T. Wu, M. Malkin, and D. Boneh. Building intrusion tolerant applications. In *Proceedings of the 8th USENIX Security Symposium*, pages 79–91, 1999.

[WVW] F. Wang, Brian Vetter, and Shyhtsun Felix Wu. Secure routing protocols: Theory and practice. Technical Report, North Carolina State University.

[XTZW05] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 46–57, New York, NY, USA, 2005. ACM Press.

[Y. 98] Y. Ko and N. H. Vaidya. Location-Aided Routing(LAR) in Mobile Ad Hoc Networks. In *The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, October 1998.

[Y. 00a] Y. Kim and A. Perrig and G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *ACM Conference on Computer and Communications Security*, pages 235–244, 2000.

[Y. 00b] Y. Zhang and W. Lee. Intrusion Detection in Wireless Ad-Hoc Networks. In *The Sixth Annual ACM/IEEE Conference on Mobile Computing and Networking*, August 2000.

[YK03] S. Yi and R. Kravets. MOCA: MObile certificate authority for wireless ad hoc networks. In *Proceedings of the 2nd Annual PKI Research Workshop (PKI 03)*, April 2003.

[YK04a] S. Yi and R. Kravets. Composite key management for ad hoc networks. In *Proceedings of The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous 04)*, August 2004.

[YK04b] S. Yi and R. Kravets. MOCA : MObile certificate authority for wireless ad hoc networks. Technical Report UIUCDCS-R-2004-2502/UILU-ENG-2004-1805, University of Illinois at Urbana-Champaign, 2004.

[YLN03] J. Yoon, M. Liu, and B. Noble. Sound mobility models. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 205–216. ACM Press, 2003.

[YNK02] S. Yi, P. Naldurg, and R. Kravets. Integrating quality of protection into ad hoc routing protocols. In *Proceedings of The 6th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 02)*, August 2002.

[Z. 97] Z. Haas and M. Pearlman. The zone routing protocol (ZRP) for ad hoc networks. Internet draft, draft-zone-routing-protocol-00.txt, 1997.

[ZH99] L. Zhou and Z. J. Haas. Securing Ad Hoc Networks. *IEEE Network Magazine*, November 1999.

[Zim95] P. Zimmermann. The official PGP user's guide. MIT Press, 1995.

[ZSvR] L. Zhou, F. Schneider, and R. van Renesse. APSS: Proactive secret sharing in asynchronous systems. In preparation.

Vita

Seung Yi graduated from Seoul National University, Seoul, Korea in 1995 with a Bachelor of Science degree in Computer Science. After graduation, he joined a start-up company, Nexon Co., as a senior research programmer, first as a part of and later as a leader of the team developing the game engine of world's first graphic massive multi-player online game, the Kingdom of the Wind. After he left Nexon in 1996, he enrolled in the graduate program at the Department of Computer Science, University of Illinois at Urbana-Champaign in 1997. He participated in several projects as a member of two research groups, System Software Research Group led by professor Roy Campbell and Mobius Group led by professor Robin Kravets. He obtained his Doctor of Philosophy Degree in October 2005. His research interests include system and network security, distributed systems, and wireless networking.